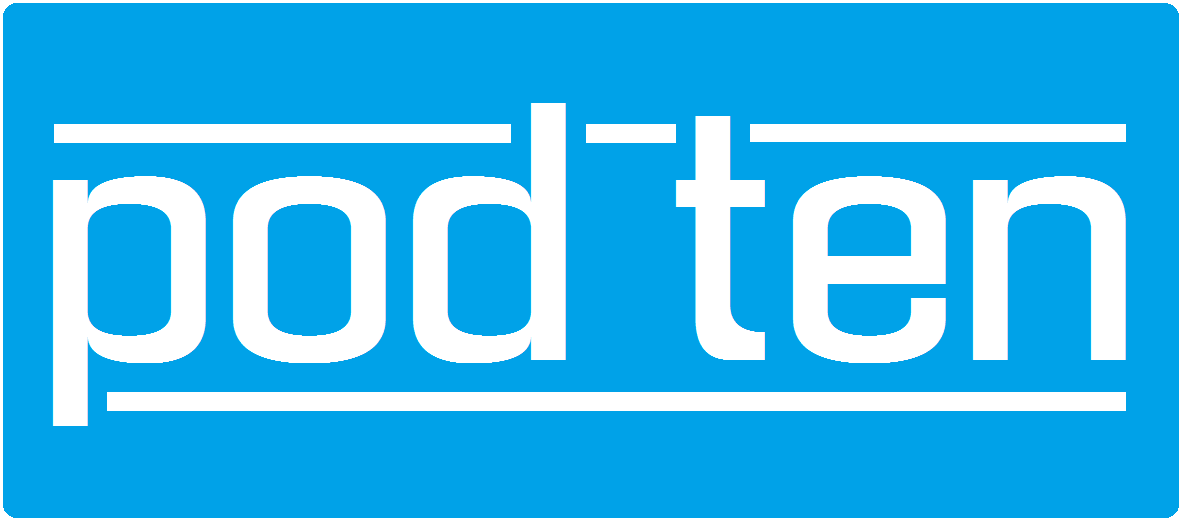
**AirPlan – Online Flight Booking System**

AirPlan Final Report

EECE 419 Assignment #5

Document Ref.: EECE419-Pod-10\_Final\_Report

First Issue- November 25, 2012



**CIRCULATION / DISTRIBUTION LIST**

|  |  |
| --- | --- |
| **NAME** | **STUDENT#** |
| Anuj Mehta | 23610082 |
| Ryan Randhawa | 62768080 |
| Kevin Chow | 44692093 |
| Nathaniel Sham | 65635088 |
| Yu Han | 43088087 |
| Louie Yuen | 75658096 |
| Yonatan Indrajaya | 42087072 |
| Scott Pearson | 43821081 |
| Shirley Gong | 63769087 |

**DOCUMENT IDENTIFICATION SHEET**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | | |
| Date of first issue: 11/01/2012 | | Current version number : Version 1.0 | |
|  | | | |
| Prepared by : | EECE 419 Pod 10 | | Date : 11/25/2012 |
|  |  | |  |
| Verified by : | Ryan Randhawa | | Date : 11/25/2012 |
|  |  | |  |
| Approved by : | Anuj Mehta | | Date : 11/25/2012 |
|  |  | |  |
| Status : | COMPLETED | | Applicability Date: 11/25/2012 |
|  | | | |

**Copyright notice**

This document may not be reproduced (even partially) or communicated to third parties without the written authorization of Pod Ten Project Management.

DOCUMENT CHANGE RECORD

|  |  |  |
| --- | --- | --- |
| **Date** | **Description** | **Affected sections** |
| 11/25/2012 | First Issue | All |
| 11/25/2012 | Version 1.0 | All |

Contents

[1. Introduction 9](#_Toc341651545)

[1.1 Purpose 9](#_Toc341651546)

[1.2 Scope 9](#_Toc341651547)

[1.3 Overview 9](#_Toc341651548)

[2. Results and Challenges Faced 10](#_Toc341651549)

[3. Glossary 12](#_Toc341651550)

[4. User Manual 14](#_Toc341651551)

[4.1 Installation Instructions and System Requirements 14](#_Toc341651552)

[4.2 Customer Manual 15](#_Toc341651553)

[4.2.1 Logging In/Creating a New User 15](#_Toc341651554)

[4.2.1.1 The Main Screen: 15](#_Toc341651555)

[4.2.1.2 Create New User Screen 15](#_Toc341651556)

[4.2.2 Navigating the Website 16](#_Toc341651557)

[4.2.2.1 Navigation 16](#_Toc341651558)

[4.2.2.2 Tabs 16](#_Toc341651559)

[4.2.3 Searching and Reserving Flights 17](#_Toc341651560)

[4.2.3.1 Searching for flights 17](#_Toc341651561)

[4.2.3.2 Reserving a Flight 18](#_Toc341651562)

[4.2.4 Manage Bookings 21](#_Toc341651563)

[4.2.4.1 Selecting a Booking to Manage 21](#_Toc341651564)

[4.2.4.2 Managing Your Booking 21](#_Toc341651565)

[4.2.4.3 Printing the Boarding Pass 21](#_Toc341651566)

[4.2.5 View Transaction History 23](#_Toc341651567)

[4.2.5.1 Loyalty Points 24](#_Toc341651568)

[4.3 Staff Manual 25](#_Toc341651569)

[4.3.1 Logging In 25](#_Toc341651570)

[4.3.1.1 The main screen: 25](#_Toc341651571)

[4.3.1.2 Staff Login Screen: 25](#_Toc341651573)

[4.3.1.3 Notification of Account Privledge Level 26](#_Toc341651574)

[4.3.2 Navigating the Website 27](#_Toc341651575)

[4.3.2.1 Navigation 27](#_Toc341651576)

[4.3.2.2 Tabs 27](#_Toc341651577)

[4.3.3 Managing User Bookings 28](#_Toc341651578)

[4.3.3.1 Accessing a Customers Bookings 28](#_Toc341651579)

[4.3.3.2 Editing Booking Information 29](#_Toc341651582)

[4.3.4 Viewing Flight Manifests 30](#_Toc341651585)

[4.3.4.1 Selecting a Flight to View the Manifest 30](#_Toc341651586)

[4.4 Management Manual 31](#_Toc341651588)

[4.4.1 Logging In 31](#_Toc341651589)

[4.4.1.1 The main screen: 31](#_Toc341651590)

[4.4.1.2 Management Login Screen 31](#_Toc341651592)

[4.4.1.3 Notification of Account Privledge Level 32](#_Toc341651593)

[4.4.2 Navigating the Website 33](#_Toc341651594)

[4.4.2.1 Navigation 33](#_Toc341651595)

[4.4.2.2 Tabs 33](#_Toc341651596)

[4.4.3 Managing User Bookings 34](#_Toc341651597)

[4.4.3.1 Accessing a Customers Bookings 34](#_Toc341651598)

[4.4.3.2 Editing Booking Information 35](#_Toc341651600)

[4.4.4 Viewing Flight Manifests 37](#_Toc341651601)

[4.4.4.1 Selecting a Flight to View the Manifest 37](#_Toc341651602)

[4.4.5 View Flight Statistics 38](#_Toc341651604)

[5. Requirements 39](#_Toc341651605)

[5.1 General Description of Requirements 39](#_Toc341651606)

[5.2 Use Case UML Diagram 39](#_Toc341651607)

[5.3 Functional Requirements 40](#_Toc341651608)

[5.4 Non Functional Requirements 41](#_Toc341651609)

[5.4.1 Security requirements 41](#_Toc341651610)

[5.4.2 Performance requirements 41](#_Toc341651611)

[5.4.3 Interface requirements 41](#_Toc341651612)

[5.4.4 Operational requirements 41](#_Toc341651613)

[5.5 Use Cases 43](#_Toc341651614)

[5.5.1 Customer Use Cases 43](#_Toc341651615)

[5.5.2 Airline Staff and Management Use Cases 49](#_Toc341651616)

[6. Architecture 53](#_Toc341651617)

[6.1 Architecture of Struts, Tools Used and Patterns Followed 53](#_Toc341651618)

[6.2 Sequence Diagrams 58](#_Toc341651619)

[6.2.1 Search 58](#_Toc341651620)

[6.2.2 Reserve 58](#_Toc341651621)

[6.2.3 View Transactions 59](#_Toc341651622)

[6.2.4 Print Boarding Pass 60](#_Toc341651623)

[6.2.5 Modify Reservation 61](#_Toc341651624)

[6.2.6 Cancel Reservation 62](#_Toc341651625)

[6.2.7 View Flight Manifest (Staff) 63](#_Toc341651626)

[6.2.8 View Flight Manifest (Management) 64](#_Toc341651627)

[6.2.9 View Flight Statistics 65](#_Toc341651628)

[6.2.10 Modify Reservation on Behalf of Customers 66](#_Toc341651629)

[6.2.11 Cancel Reservation on Behalf of Customers 67](#_Toc341651630)

[6.3 Deployment Architecture 68](#_Toc341651631)

[7. Interesting Design Aspects 69](#_Toc341651632)

[7.1 User Authentication and Security 69](#_Toc341651633)

[7.2 Front-end engineering 69](#_Toc341651634)

[7.3 Other Features 70](#_Toc341651635)

[Appendix A: Test Cases and Test Scripts 72](#_Toc341651636)

[Customer Use Cases 72](#_Toc341651637)

[Search Flight 72](#_Toc341651638)

[Make a Reservation 75](#_Toc341651639)

[View Transaction History 77](#_Toc341651640)

[Modify Reservation 79](#_Toc341651641)

[Cancel Reservation 82](#_Toc341651642)

[View and Print Boarding Pass 84](#_Toc341651643)

[Staff and Management Use Cases 87](#_Toc341651644)

[View Flight Manifest (Airline Staff) 87](#_Toc341651645)

[View Flight Manifest (Airline Management) 90](#_Toc341651646)

[View Flight Statistics 94](#_Toc341651647)

[Modify Reservation on Behalf of Customer 96](#_Toc341651648)

[Cancel Reservation on Behalf of Customer 97](#_Toc341651649)

[Non Functional Requirements 98](#_Toc341651650)

[User Login Cases 101](#_Toc341651651)

[Unit Tests 104](#_Toc341651652)

[User Service Tests 104](#_Toc341651653)

[Reservation Service Unit Tests 107](#_Toc341651654)

[Flight Service Unit Tests 110](#_Toc341651655)

[Appendix B: Box 112](#_Toc341651656)

[Appendix C: Breakdown of workload 114](#_Toc341651659)

[Appendix C: Code 125](#_Toc341651660)

# Introduction

## Purpose

The purpose of this document is to describe the finished AirPlan flight reservation and management Tool. This includes an overview of the result of the project along with the challenges we faced while developing it, the user manual describing how to use the project, the software requirements specification which outlines the required functionality of the application, the underlying architecture on which the entire application was built, the test cases and scripts which were used to verify the correctness of the code, and an analysis of the design and architecture of the project. This report is intended as documentation for other developers wishing to add to or modify the application, and end users who wish to use it.

## Scope

The rest of this document is split into several sections. Included in this document are the following: an overview of the results of the project and the challenges we faced, the user manual, the software requirements specification, the architecture, test cases and scripts, and a description of some of the interesting design aspects of the project. This report starts with describing the outcome of the project, the success as seen in our team’s implementation, and the challenges we faced during the term. The user manual provides step by step instructions to all the users of AirPlan- customers, staff and management. The software requirements specification outlines the required functionality of the application and how it is meant to be used in the form of use cases. The architecture describes the general framework of the application, which all subsequent work was based on. Test cases have been included in the appendix to give specific conditions and inputs to various parts of the program and their expected results. The appendix also includes a breakdown of the workload throughout the term, and a snapshot of the latest source code of the project.

## Overview

AirPlan is an online flight reservation and management system for airline customers and staff. It provides an intuitive interface which supports functionality for customers such as searching for flights, reserving seats on a flight, managing bookings and payments, and printing boarding passes. Additionally, it allows airline staff and managers to view flight manifests and flight statistics as well as to manage customer bookings on their behalf. AirPlan provides detailed information about the bookings to airline staff such as seat placement and meal preferences. Managers can view additional useful information such as the percentage of flights delayed, the percentage of flights that were full, the percentage of flights that were more than half empty, seat availability information and destination specific information. The goal of AirPlan is to provide complete and efficient functionality for customers booking, paying for, and managing flights.

AirPlan was written in the Java programming language, and is hosted on an Apache Tomcat web server. AirPlan is best viewed using the latest version of Google Chrome.

# Results and Challenges Faced

After three months of work, Pod Ten has successfully taken a proposal and converted into a fully functional product. Pod Ten defined a detailed description of each feature of the product, designed a scalable architecture for the overall system, created an exhaustive set of test cases and scripts and implemented this system to make a great product.

AirPlan is now a fully functional flight reservation and management system. Customers are able to search for and reserve flights in an intuitive way through an easy to use web application. They are also able to view their past transactions, print their boarding passes and modify and cancel reservations through this web application. Airline staff and management are able to view flight manifests and help customers modify and cancel their reservations. Airline management is also able to view essential flight statistics through a real time dashboard.

These successes did not come without its set of challenges. At the start of the term, very few members of the team had any experience at all in working on such large projects. Even fewer had the experience of working on a web product.

One of the problems the team faced early on was learning all the new technologies we were using to build this project. We decided that due to the time constraints we should avoid building our own web framework, so we went with something that was available freely and trusted. Learning to develop within our chosen technologies proved to be a challenge, and as time progressed, it became apparent that even coding styles and conventions had to be established.

Apart from learning to develop within the web framework, team members also had to learn concepts such as web servers, databases and hosting servers. It was initially difficult to understand how requests were passed from the web server to the web framework and how results were passed to a templating engine that would then product HTML responses. Some of our team members had not even learnt about databases, so it was new to them that data had to be persisted into a database, and not just stored in memory. All of this led to a steeper learning curve.

We were also initially concerned with the amount of time documentation was taking. We thought that this was just additional overhead that we just had to deal with, but we soon realised that this overhead was worth it. Keeping track of changing requirements, and keeping the entire team in the loop about architectural changes proved to be a difficult task, and the documents that we wrote were the best resources to help us with it.

Despite all these challenges, AirPlan is quite a success. Not only is the product feature complete, but it has also been executed very well. The user interface is beautiful and intuitive, the website is very responsive, and great care has been taken in designing the user experience to ensure that users are never confused while using the product. The architecture of the system is scalable and easily maintainable, the code is separated into several layers- each with its own function, and the database has been designed to be very efficient. All of this has helped make AirPlan a great product, something that the entire team would like to see being used in the real world, if not for the nature of the product.

# Glossary

**Bill/Billing** - A financial transaction between the customer and AirPlan, for the customer to pay for a booking. A billing entry consists of the amount paid, the booking for which the bill is for, the credit card number, and whether loyalty points were used.

**Boarding Pass-** A boarding pass is a document provided to users during check in. It gives the passenger permission to board a particular flight at a particular location. A customer, on printing a boarding pass gets awarded any associated loyalty points. Also, after printing a boarding pass, that booking cannot be modified or cancelled.

**Booking** – A confirmation that the customer has paid for a set of flights. Bookings consist of tickets making up a flight route. A Booking can have multiple tickets. Each ticket may involve the same or a different passenger. For example: if a father wants to book a vacation to San Francisco and Los Angeles for his entire family, he may book multiple tickets for both destinations, all under one booking.

**Customer** – The primary users of the AirPlan reservation service. They are able to search for and reserve flights, view bookings, and manage payments/loyalty points. They can also view their financial transactions on the site. They do not have access to the back end portions of the system such as viewing flight manifests or flight statistics. They are sometimes referred to as Users.

**Flight** – A flight transports passengers from the flight’s origin to its destination. Passengers must purchase tickets to board Flights. Flights have some additional properties such as estimated departure and arrival times, actual departure and arrival times, numbers of economy/business class seats available, cost, and airline name.

**Flight Manifest** – A list of tickets for that flight, listing all the customers who will be aboard that flight, and information about their seating, meal preferences, etc.

**Loyalty Points** – Customers are given loyalty points when they pay for a flight without using loyalty points. Loyalty points can be used in addition to credit cards when paying for a Booking.

**Manager** – Staff who have elevated privileges. They may do anything staff can do including viewing flight manifests and managing customer bookings. They are provided additional functionality such as being able to view detailed flight statistics, and viewing all flight manifests, not just the ones from the airport they belong to.

**Place** – An airport location. It is used to describe the origin and destinations of flights as well as to indicate where each staff member is based out of.

**Staff** – Primary users of the AirPlan flight management system. Their purpose is to ensure that customers are able to use this system effectively, and that their time flying is pleasant and comfortable. Each staff member is assigned an airport that they are based out of (examples: Vancouver, Ottawa etc). They can manage bookings on behalf of all customers, but they can only view the flight manifests of those flights either originating from or arriving at their home location. They may not view any flight statistics.

**Ticket** – A ticket represents a customer’s proof that he has paid for a particular flight for a particular passenger. Each ticket is only valid for one specific flight from one origin to one destination. However, to board that flight, the customer must print a boarding pass for that flight. Users can do so by using the print boarding pass functionality offered by AirPlan. Other information on the ticket includes the seat number and type, and meal preferences. Tickets belong to a Booking, and there can be multiple tickets in a Booking.

**User** – Any user of the AirPlan flight management and reservation system. This includes customers, staff and management.

# User Manual

## Installation Instructions and System Requirements

AirPlan is a web application hosted on a remote web server. It is accessible via any web browser; however we have chosen to provide full support only to the users of Google Chrome. Thus, to use this software, users need not install anything but a web browser on their computers. To install Chrome, users must satisfy the following hardware and software requirements:

* Google Chrome version 21 or higher
* 250 MB free disk space
* 512 MB RAM
* An AirPlan account (free to create)
* A valid email address (for all correspondence)
* Access to a printer (for boarding passes)

In addition, computers should meet the minimum operating system and processor requirements as listed in Table 1:

|  |  |
| --- | --- |
| **Minimum Operating System Version** | **Minimum Processor** |
| Windows XP Service Pack 2 | Intel Pentium 4 or later |
| Mac OS X 10.5.6 | Intel Processor |
| Ubuntu 10.04+  Debian 6+  Fedora Linux 14+ | Intel Pentium 3 or AMD Athlon 64 |

Table 1: Operating system and processor requirements for Google Chrome.

## Customer Manual

## Logging In/Creating a New User

### The Main Screen:

All users of AirPlan must be logged in to use the website. To login, you must simply go to the home page and enter their email address and their password.

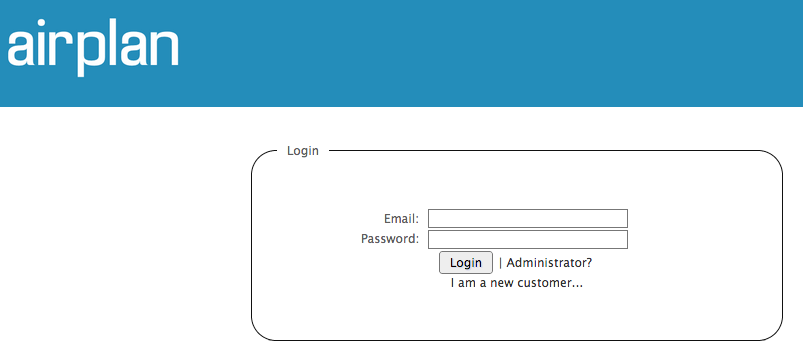


Figure 1: Login

### Create New User Screen

To create a new account with AirPlan, click on the ‘I am a new customer’ link and fill out all the fields. Once you are done click on the ‘Create’ button.

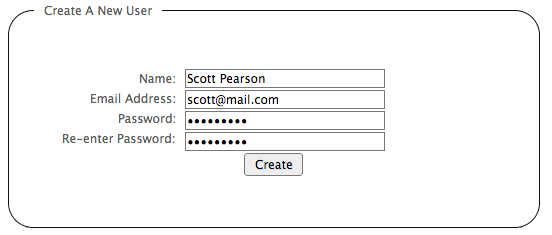


Figure 2: Create New User

Congratulations, you have now successfully created an account with AirPlan!

## Navigating the Website

### Navigation

AirPlan uses tabs to allow for easy navigation. These tabs are found at the top of the page - in the header - and are available from every screen once logged in.

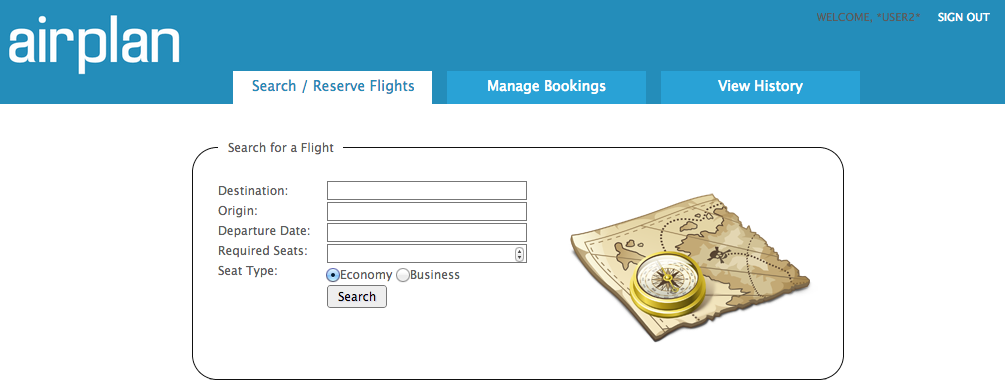


Figure 3: Navigation

### Tabs

To make navigation simple, the tabs are named in a clear and direct manner and are covered in detail in further sections of the user manual.

*Search / Reserve Flights:* Search for flights by defined input criteria, the flights returned may then be reserved. For more information on searching and reserving flights please see: ‘Section 3 - Searching and Reserving Flights’.

*Manage Bookings:* Once flights have been reserved; allows the user to: print boarding passes, modify ticket details, and cancel bookings. For more information on managing bookings please see: ‘Section 4 – Managing Bookings’.

*View History:* A complete history of all flights booked with AirPlan, including loyalty point updates and total points amount.

For more information on viewing flight bookings history please see: ‘Section 5 - View History’

For more information on loyalty points please see: ‘Section 6 – Loyalty Points’.

## Searching and Reserving Flights

### Searching for flights

To search for a flight, click on the ‘Search / Reserve Flights’ tab.

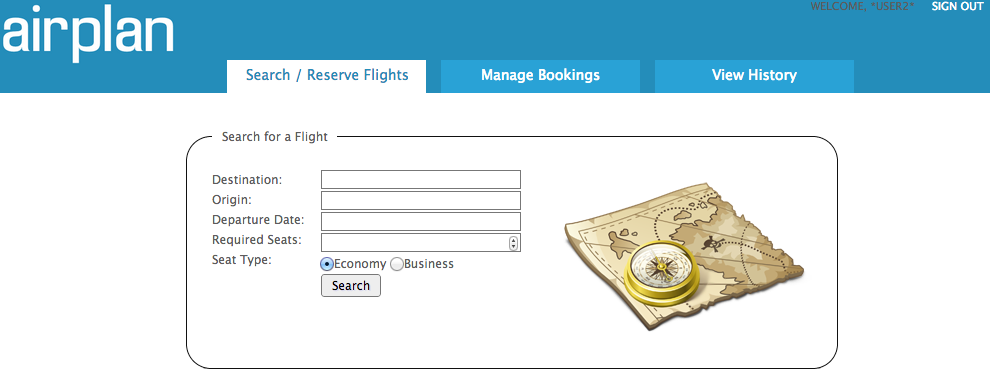


Figure 4: Search Flights

To designate the destination and origin of your flight: begin to type desired city location and a menu will be generated with entries matching the input city. Using the arrow keys, or mouse, please select the desired location from this menu.

Departure date may be specified with the built in calendar menu – available by clicking in the departure date input field – or entering the date by text in the form of: ‘YYYY-MM-DD’.

The required seats section allows you to book multiple seats on the same flight at once. Lastly, seat type may be chosen to be ‘Economy’ or ‘Business’.

For example, Figure 5 shows how you can book a flight from Chicago to Vancouver on Nov, 27, 2012.



Figure 5: Search details

Clicking on ‘Search’ will then display the search results as seen in Figure 6.

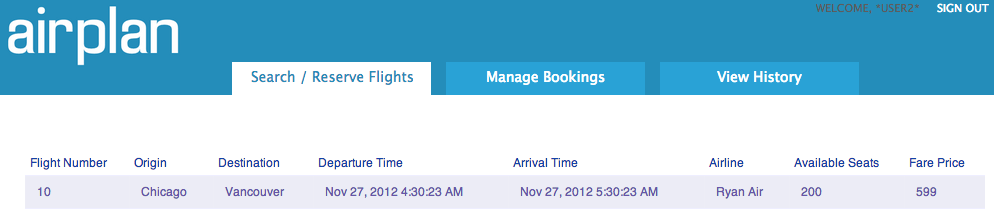


Figure 6: Search Results

Matching flight information is displayed and a flight can be selected by clicking on it. The flight information includes: the flight number (flight designating number unique to each flight), origin, destination, departure time, arrival time, the name of the airline operating the flight, seats available on the flight, and the price of the flight.

### Reserving a Flight

Selecting a flight from the search results will allow you to reserve one or more seats on the flight. Here you may specify reservation details and preferences. Updating your flight preferences is accomplished by accessing the Manage Booking section covered in ‘Section 4 – Manage Bookings’.

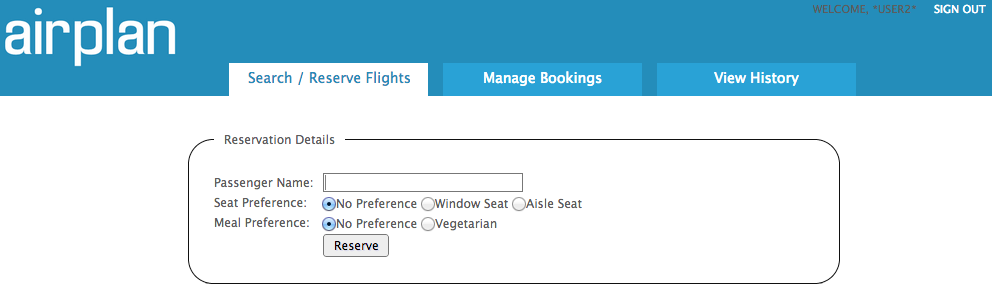


Figure 7: Reserve Flight

Please Note- the passenger name input field should contain the name to be printed on the ticket.

Once you have specified the passenger name and your preferences click ‘Reserve’. This will direct you to the reservation confirmation page. Please review the flight information before continuing to checkout.

As seen in Figure 8, the next page gives you an option to add another flight to your shopping cart, or to add a return flight to your shopping cart.

The ‘Add another flight’ button will return you to the search for flights screen. The ‘Find Return Flight’ button will also return you to the search for flights screen with the destination and origin of your flight interchanged.

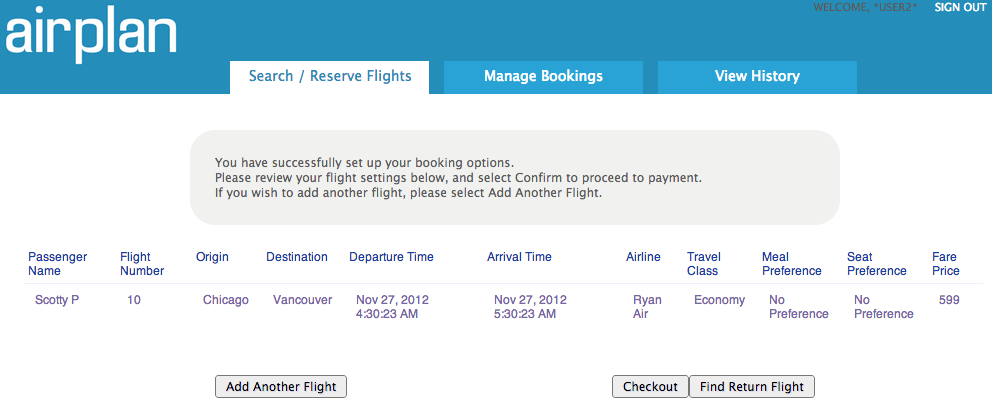


Figure 8: Review Flight Settings

Checking out with the ‘Checkout’ button directs you to the final stage of the reservation process- the payment page. Figure 9 below shows you an example of what the checkout page should look like.

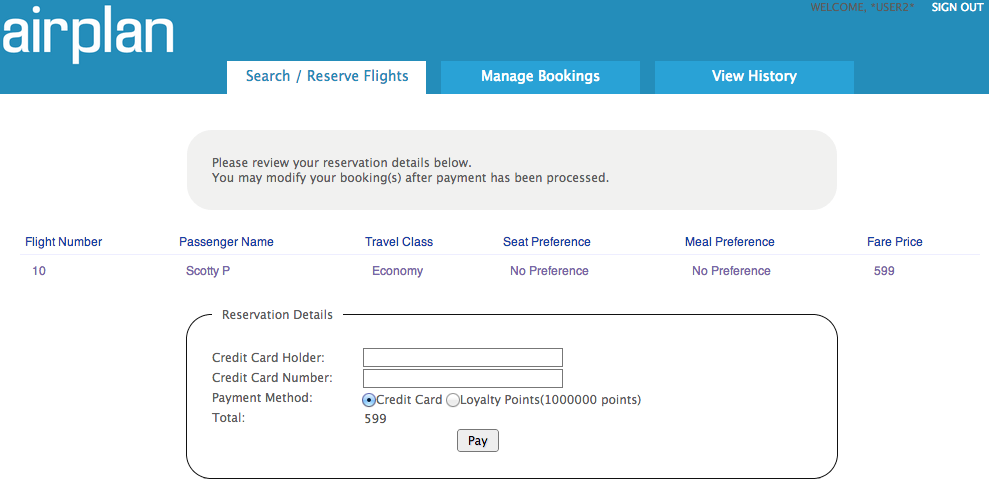


Figure 9: Checkout

Please review your flight details again before proceeding with the payment. There are two payment options: credit card and loyalty points (please see: ‘Section 6 – Loyalty Points’ for more information on the loyalty points system).

To pay with a credit card select ‘Credit Card’ as the payment method, and enter the card details in the provided fields.

To pay with points select ‘Loyalty Points’ as the payment method – your total available points are displayed and must be greater than the cost of the flight in dollars to redeem them for the payment. The value of points is equivalent to that of dollars for payments with AirPlan.

Upon completion, a summary of your payment is then displayed (as seen in Figure 10).

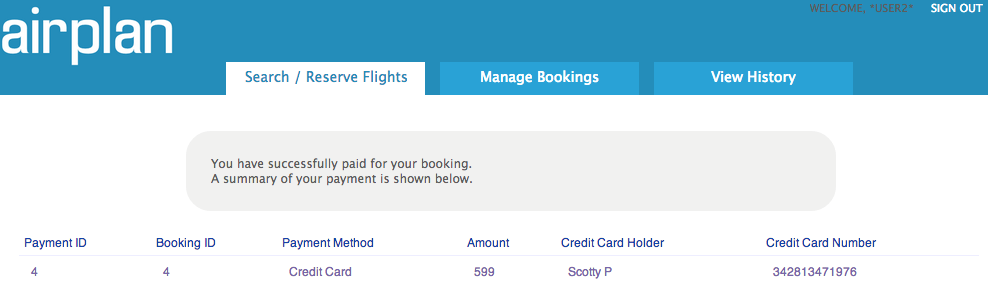


Figure 10: Summary of Reservation

Please note- this reservation is not finalized until the boarding pass has been printed. Please see: ‘Section 4.3 –Printing the Boarding Pass’ for more information and instructions to print the boarding pass.

## Manage Bookings

### Selecting a Booking to Manage

From the ‘Manage Bookings’ tab you may: print your boarding pass, modify a ticket, or cancel the booking. Bookings are sorted by the date the booking is created.

Select the booking you wish to manage by clicking on its Booking ID.

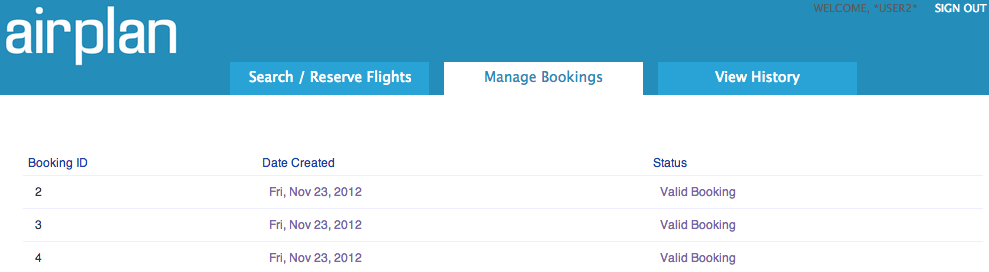


Figure 11: Manage Bookings

### Managing Your Booking

On clicking on the booking Id, you will be directed to the Booking Details page. This page presents the bookings details of that booking along with several options.

You may click on ‘Cancel Booking’ to remove the booking from your account, you can change the preferences of your booking by clicking on the ‘Modify Ticket’ link, and you can print the boarding pass by clicking on the ‘Print Boarding Pass’ link. You may or may not be able to cancel or modify your booking depending on if you have already printed the boarding pass. Once a boarding pass has been printed, loyalty points are granted to the user for that booking, and no further changes to it are possible.

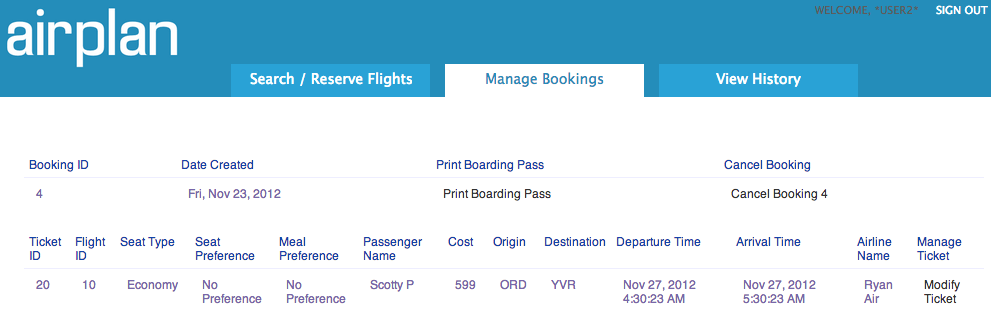


Figure 12: Manage a particular booking

### Printing the Boarding Pass

Printing the boarding pass is crucial; you will need this pass to board the airplane.

From the ‘Manage Bookings’ tab once you have selected the flight you wish to print the boarding pass for, click ‘Print Boarding Pass’.

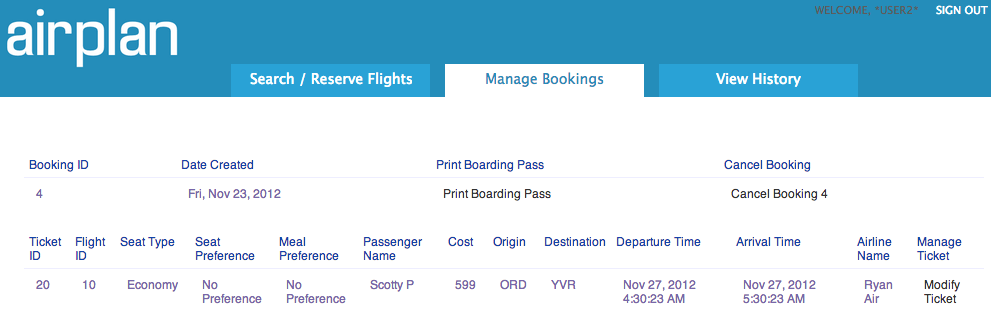


Figure 13: View Boarding Pass

You will then be directed to your boarding pass, which you must print from your browser. In Google Chrome, you may click on the File menu and click on the Print option, as shown in Figure 14.

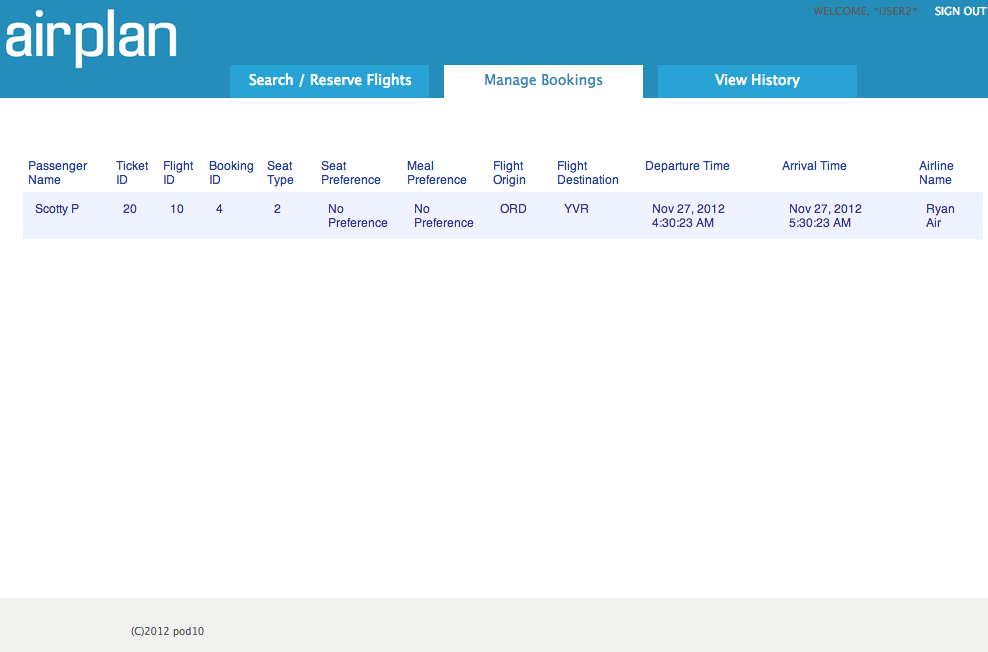
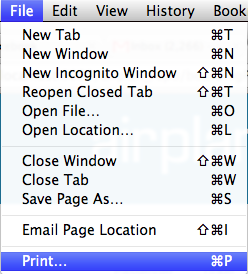


Figure 14: Print Boarding Pass

After the boarding pass is printed, the flights status in ‘Manage Bookings’ will be updated from ‘Valid Booking’ to “Boarding Pass Printed’.

## View Transaction History

The ‘View History’ tab displays a complete history of all your transactions. From this page you may view your account activity or track your accumulation and use of loyalty points. Loyalty points are explained in the next section.

There will be at least one entry for each booking made. If you cancel a booking, another entry will be made showing the cancellation. There will also be entries for when you are awarded loyalty points for a past booking.

Note: View History relates to transactions; here name displays the name on the credit card.

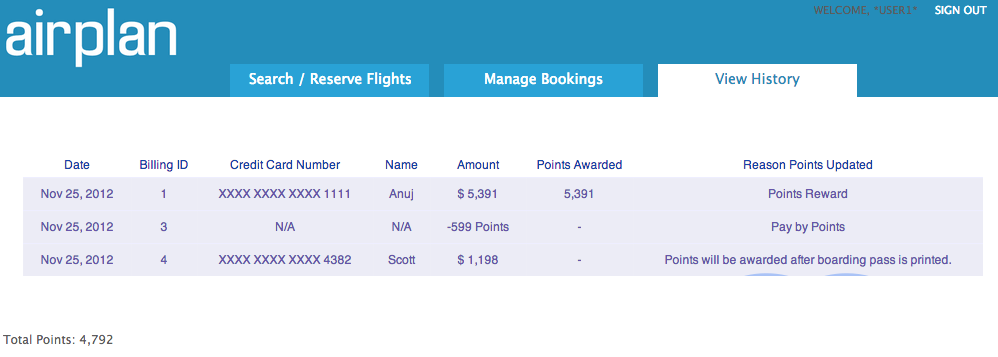


Figure 15: View Transaction History

## Loyalty Points

Loyalty points are our way of rewarding our loyal customers.

Loyalty points are awarded for flights booked with AirPlan using a credit card. Points may be redeemed to pay for flights at the rate of one point per dollar value of the flight- that is, if a flight costs $599 dollars, a customer could instead redeem 599 points to pay for the flight.

Loyalty points are only awarded once the booking’s boarding pass has been printed, and you can view a summary of your points from the ‘View History’ tab.



Figure 16: Loyalty Points

Total points is shown on the bottom left, the amount of points awarded per flight and the reason they have been updated is also shown. Here no points have been awarded yet for the flight because the boarding pass has not been printed. Points can also be updated for a flight to show a deduction of points, for the redeeming points for a free flight.

## Staff Manual

### Logging In

## The main screen:

To enter the account login with staff privileges screen, please select the ‘Administrator?’ option from the main screen.

## 

Figure 17: Staff Login

## Staff Login Screen:

Please input you email and password provided your company into the appropriate input fields.

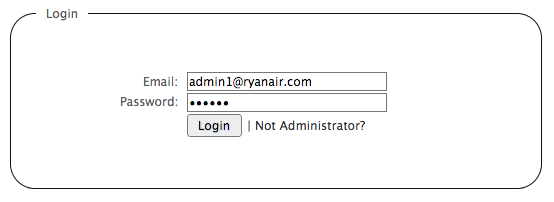


Figure 18: Staff Login 2

## Notification of Account Privledge Level

After loging in you will be directed to the account privledge level page which will notify you of your account privledges:

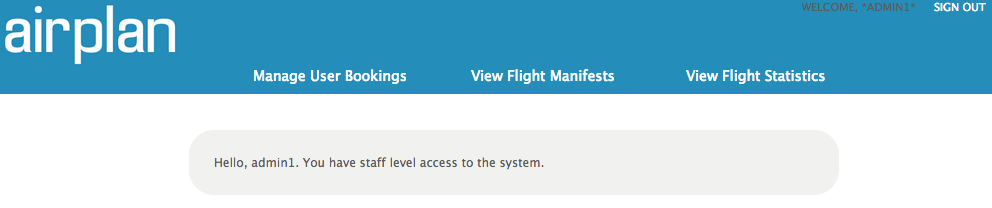


Figure 19: Staff Notification of Priviledges

## Navigating the Website

### Navigation

AirPlan uses tabs to allow for easy navigation. These tabs are found at the top of the page - in the header - and are available from every screen once logged in.

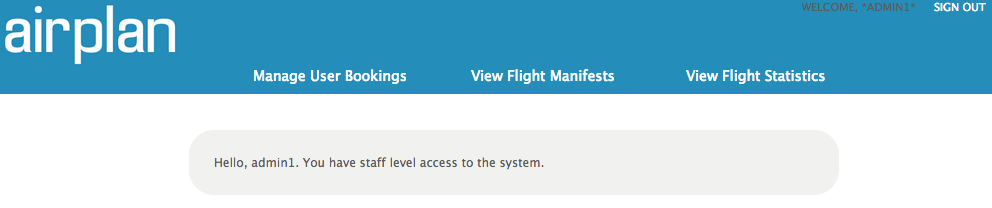


Figure 20: Staff Navigation

### Tabs

To make navigation simple, the tabs are named in a clear and direct manner and are covered in detail in further sections of the user manual.

*Manage User Bookings:* Manage bookings on behalf of a user. For more information on managing bookings for users please see: ‘Section 3 – Managing User Bookings’

*View Flight Manifest:* Staff are able to view the flight manifests for flights leaving from or going to the airport they work at. For more information on viewing flight manifest please see: ‘Section 4 – Viewing Flight Manifests’

*View Flight Statistics:* Shows statistics including the amount of flights into and from airports, and the most popular flight destination. Also displayed are the amounts of flights: early, delayed, on time, full, and half empty. For more information on viewing flight statistics please see: ‘Section 5 - View Flight Statistics’

## Managing User Bookings

### Accessing a Customers Bookings

To retrieve a customers booking, click on the ‘Manage User Bookings’ tab. Next, input the email they provide you with and select ‘Retrieve Bookings’.

### 

Figure 21: Staff Manage Bookings

Clicking ‘Retrieve bookings’ will then display the search results.

### 

Figure 22: Staff Manage Bookings 2

Select the booking to edit by clicking its Booking ID Number.

### Editing Booking Information

The booking selected in section 3.1 will be displayed in the following format:

### 

### 

Figure 23: Staff Edit Booking

Here you may cancel the booking on behalf of the customer with the ‘Cancel Booking #’ button, or edit ticket details with the ‘Modify Ticket’ button. Modifying tickets allows for updating the reservation details - the customer name, seat preference, and meal preference.

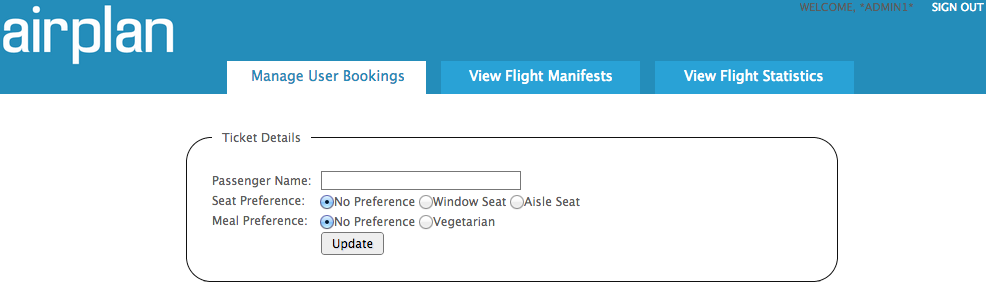


Figure 24: Staff Edit Booking 2

Note: the passenger name is the name to be printed on the ticket.

## Viewing Flight Manifests

### Selecting a Flight to View the Manifest

From the ‘View Flight Manifests’ tab you may: select to view the flight manifest of any flight where the origin or destination is your location by clicking on its Flight ID Number.

### 

Figure 25: Staff View Flight Manifest

This will then display the flight manifest for the selected flight.

## Management Manual

## Logging In

### The main screen:

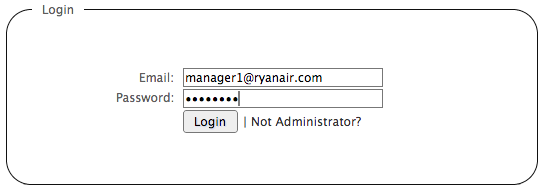
To enter the account login with manager privileges screen, please select the ‘Administrator?’ option from the main screen.

### 

Figure 26: Management Login

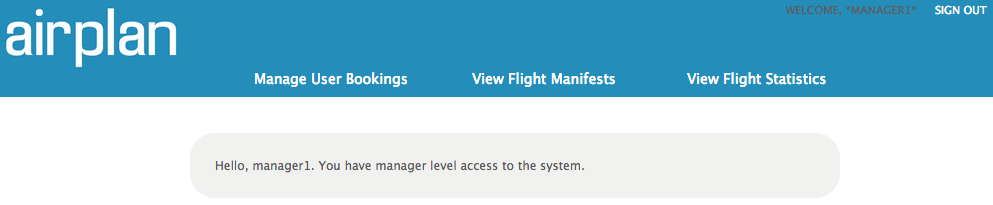
### Management Login Screen

Please input you email and password provided your company into the appropriate input fields.

Figure 27: Management Login 2

### Notification of Account Privilege Level

After logging in you will be directed to the account privledge level page which will notify you of your account privledges:

Figure 28: Management Notification of Privileges

## Navigating the Website

## Navigation

AirPlan uses tabs to allow for easy navigation. These tabs are found at the top of the page - in the header - and are available from every screen once logged in.

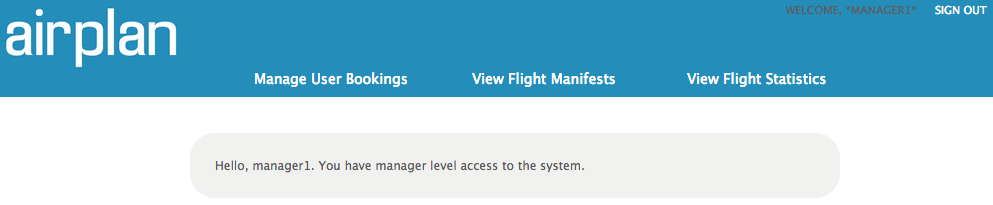


Figure 29: Management Navigation

## Tabs

To make navigation simple, the tabs are named in a clear and direct manner and are covered in detail in further sections of the user manual.

*Manage User Bookings:* Manage bookings on behalf of a user. For more information on managing bookings for users please see: ‘Section 3 – Managing User Bookings’

*View Flight Manifest:* Managers may view the flight manifests for all flights. For more information on viewing flight manifest please see: ‘Section 4 – Viewing Flight Manifests’

*View Flight Statistics:* Shows statistics including the amount of flights into and from airports, and the most popular flight destination. Also displayed are the amounts of flights: early, delayed, on time, full, and half empty. For more information on viewing flight statistics please see: ‘Section 5 - View Flight Statistics’.

## Managing User Bookings

### Accessing a Customers Bookings

To retrieve a customers booking, click on the ‘Manage User Bookings’ tab. Next, input the email they provide you with and select ‘Retrieve Bookings’.

### 

Figure 30: Management Manage Customer Bookings

Clicking ‘Retrieve bookings’ will then display the search results.

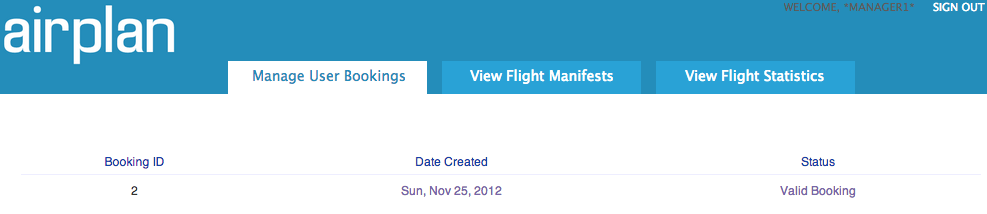


Figure 31: Management Manage Customer Bookings 2

Select the booking to edit by clicking its Booking ID Number.

### Editing Booking Information

The booking selected in section 3.1 will be displayed in the following format:

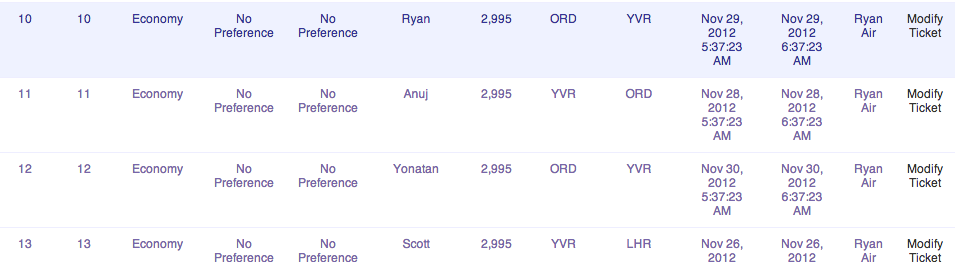
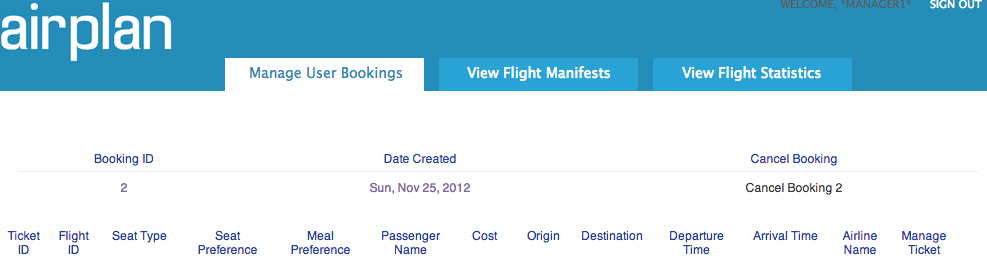


Figure 32: Management Edit Customer Bookings

Here you may cancel the booking on behalf of the customer with the ‘Cancel Booking #’ button, or edit ticket details with the ‘Modify Ticket’ button. Modifying tickets allows for updating the reservation details - the customer name, seat preference, and meal preference.

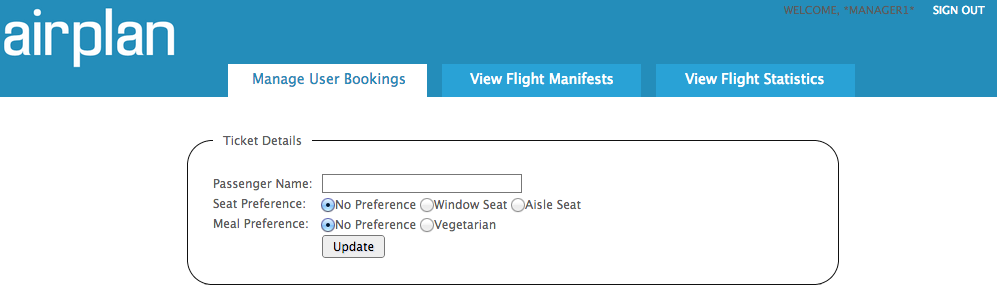


Figure 33: Management Edit Customer Booking 2

Note: the passenger name is the name to be printed on the ticket.

## Viewing Flight Manifests

### Selecting a Flight to View the Manifest

From the ‘View Flight Manifests’ tab you may: select to view the flight manifest of any flight by clicking on its Flight ID Number.

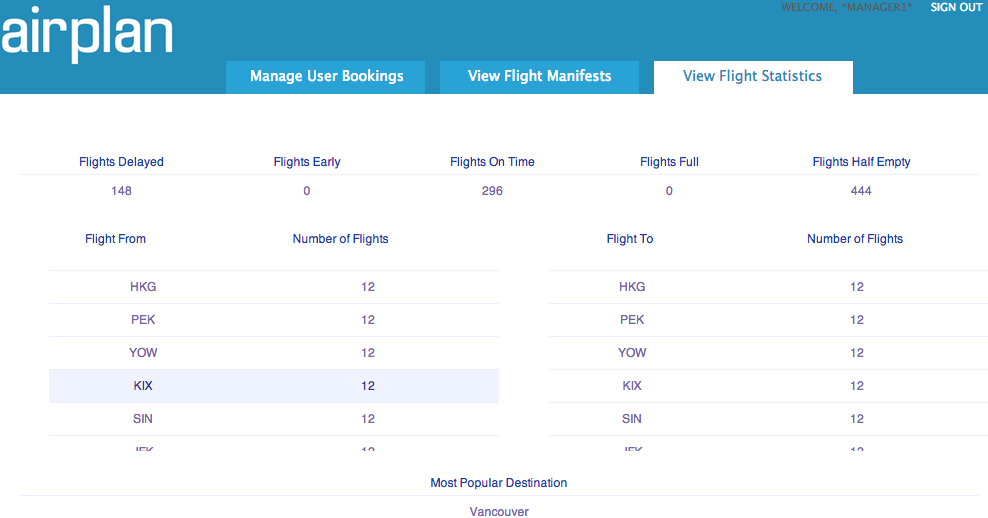
### 

Figure 34: Management View Flight Manifest

This will then display the flight manifest for the selected flight.

## View Flight Statistics

The ‘View Flight Statistics’ tab displays the following flight statistics:

Figure 35: Management View Flight Statistics

# Requirements

## General Description of Requirements

AirPlan is a flight reservation and management system that allows customers to search for and reserve flight tickets through an intuitive web interface and airline staff and management an easy to use flight management tool. The system will allow passengers to search for flights, reserve seats and buy tickets, select flight preferences, print boarding passes. It will also allow passengers to modify reservations and manage loyalty points. The airline staff and management will be able to view the manifest of each flight along with each passenger’s seating location and flight preferences. The management will also have access to statistical information such as the percentage of flights delay, the percentage of flights that were full, the percentage of flights that were more than half empty, seat availability information and destination specific information. Overall, AirPlan aims to be a one-stop shop for all flight reservation and management issues.

## Use Case UML Diagram

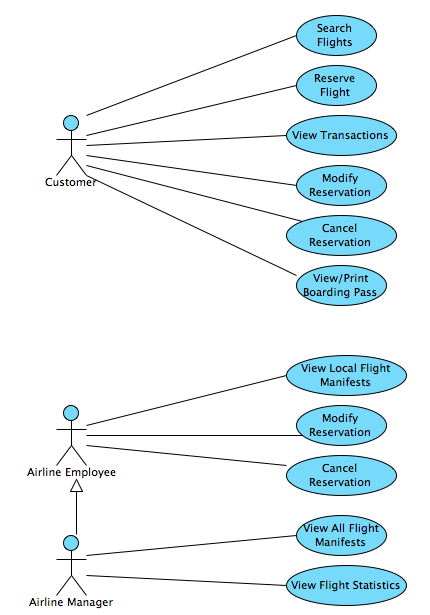


Figure 36: Use Case UML Diagram

## Functional Requirements

|  |  |
| --- | --- |
| **SR. Id.** | **Software requirements description** |
|  | **Customer Requirements** |
| F.000 | The customer shall be able to search for a flight or a set of flights by inputting required information such as from and to locations and date of travel, and optional information such as return flight date. |
| F.001 | The customer shall be able to make flight reservations of one or more flights using either their credit cards or loyalty points. |
| F.002 | The customer shall be able to manage payments by viewing all of their past transactions and checking the status of their loyalty points. |
| F.003 | The customer shall be able to logon to modify tickets for their upcoming flights. The customer shall be able to update their ticket preferences such as seat location and meal type. |
| F.004 | The customer shall be able to cancel a booking and all of the tickets associated with that booking. |
| F.005 | The customer shall be able to logon to view and print the boarding passes associated with bookings. |
|  | **Airline Staff and Management Requirements** |
| F.100 | The airline staff shall be able to view the manifest for each flight that they have access to, including seat locations and in flight preferences for each passenger. |
| F.101 | The airline management should be able to view the flight manifests of each flight, including seat locations and in flight preferences for each passenger. |
| F.102 | The airline management shall be able to view real time statistics for each flight. |
| F.103 | The airline staff and management shall be able to help customers modify their tickets for upcoming flights. |
| F.104 | The airline staff and management shall be able to help customers cancel their reservations for upcoming flights. |

Table 2: List of Functional Requirements

## Non Functional Requirements

### Security requirements

|  |  |
| --- | --- |
| **SR. Id.** | **Software requirements description** |
| SEC.0 | The system shall store all user data in secure data centers and no unauthorized access shall be granted to anyone outside or within the company.   * User data shall be stored in external databases which shall only be accessible to authorized team members. * Only data that is absolutely essential shall be requested of the user. * User data shall not be sold to external parties and shall not be used to display targeted advertisements. |

Table 3: List of Security Requirements

### Performance requirements

|  |  |
| --- | --- |
| **SR. Id.** | **Software requirements description** |
| PER.0 | No user action shall have a response time of more than 1 second.   * Users shall be notified in advance if an action is known to have large processing times. |

Table 4: List of Performance Requirements

### Interface requirements

|  |  |
| --- | --- |
| **SR. Id.** | **Software requirements description** |
| INT.0 | The product shall be web based and platform independent.   * The product shall be accessed via a web browser such as Google Chrome on any platform. This includes but is not limited to Windows, OSX and Linux. |

Table 5: List of Interface Requirements

### Operational requirements

|  |  |
| --- | --- |
| **SR. Id.** | **Software requirements description** |
| OPE.0 | All significant transactions initiated by the user shall be logged.   * Each log shall include user id, timestamp and a verbal explanation of the action. Logs may also include prior values and new values of objects. * Read and write access to production logs shall only be given to authorized users. * Logs shall be verbose enough to support investigation of bugs. * Logs shall be written for any action that involves an update to the database. These actions include but are not limited to flight reservations, loyalty point updates, payments, modifications to reservations etc. |
| OPE.1 | The consumer as well as airline employee facing applications will have 99.9% uptime between 6:00AM and 1:00AM Pacific Standard Time.   * Maintenance shall only be done between 1:00am and 6:00am Pacific Standard Time, and visitors to the website shall be notified of this maintenance at least 24 hours in advance. * The integrity of each transaction shall be protected with mechanisms such as fallback and retry etc. |

Table 6: List of Operational Requirements

## Use Cases

### Customer Use Cases

#### Use Case 1 – Search Flight (F.000)

##### Description

This use case describes how a customer can search for flights over the Internet.

##### Pre-Conditions

Customer has created a valid account.

Customer has logged into the system.

Customer’s identity has been verified.

Customer’s current login session is valid.

##### Post-Conditions

**Successful Post-Condition**:

Customer has found the most suitable flights for his/her trip.

**Failed Post-Condition**:

Customer has failed to find a suitable flight because the location is out of scope.

##### Flow of Events

1. Customer selects a starting location.
2. Customer selects a destination.
3. Customer selects a date.
4. Customer enters a number equal to the amount of passengers wishing to reserve.
5. The system shows a list of flight choices with greater or equal number of available seats than customer’s input.
6. Customer may select the flight s/he is interested in.
7. End use-case.

##### Special Requirements

Customer can stop the search and go back to the previous step at any time.

##### Extension Points

None.

#### Use Case 2 – Reserve Flight (F.001)

##### Description

This use case describes how a customer can reserve flights over the Internet.

##### Pre-Conditions

Customer has created a valid account.

Customer has logged into the system.

Customer’s identity has been verified.

Customer’s current login session is valid.

Customer has found the flights s/he wishes to reserve.

##### Post-Conditions

**Successful Post-Conditions**:

Customer has reserved desired flights with preferred class.

Customer has selected preferred seat option on the plane.

Customer has selected a preferred meal plan.

Customer has paid for the booking (using credit cards or loyalty points).

**Failed Post-Condition:**

Customer has failed to book a ticket because there are insufficient seats.

##### Flow of Events

1. Customer selects a flight and proceeds to personalization.
2. Customer enters a passenger’s name
3. Customer selects from choice of seats.
4. Customer selects from choice of meal plans.
5. Customer finalizes the booking.
6. The system displays an invoice to the customer and redirects the customer to billing module.
7. The system displays payment method and information.

*The system will only enable the option of paying by loyalty points if the user has sufficient points in their account.*

1. Customer chooses to pay with credit card and enters credit card information.
2. Customer completes the payment transaction.
3. The system updates the booking as paid, reserves the seats, and makes the ticket available to the customer.

*The system updates the manifest of the flight and meal preference list.*

1. End use-case.

Alternative A:

1. Customer chooses to pay with loyalty points.
2. The system deducts the amount from the customer’s loyalty points balance.
3. The system notifies the customer that no points will be awarded, and system updates the booking as paid, reserves the seats, and makes the ticket available to the customer.
4. End use-case.

##### Special Requirements

Customer can stop the booking process at any time before it is confirmed (step 8).

No changes can be made to the manifest until the customer makes the booking (step 9).

No loyalty points are issued at this stage (loyalty points are awarded after customer views the boarding pass).

##### Extension Points

None.

#### Use Case 3 – View Transaction History (F.002)

##### Description

This use case describes how customers can view all of their past transactions and check their loyalty point balance.

##### Pre-Conditions

Customer has created a valid account.

Customer has logged into the system.

Customer’s identity has been verified.

Customer has made reservations in the past.

##### Post-Conditions

Customer has viewed his/her past transactions.

Customer has viewed his/her loyalty points.

##### Flow of Events

1. Customer selects View Past Transactions from menu.
2. The system displays a list of transactions
3. The system displays the most updated balance of loyalty points.

*Pending loyalty points are not included in the balance (pending loyalty points are points to be gained from reserved flights after the customer has printed the boarding pass).*

##### Special Requirements

None.

##### Extension Points

None.

#### Use Case 4 – Modify Booking (F.003)

##### Description

This use case describes how a customer can modify tickets associated with a booking over the Internet.

##### Pre-Conditions

Customer has created a valid account.

Customer has logged into the system.

Customer’s identity has been verified.

Customer has reserved a ticket and has a confirmed booking.

##### Post-Conditions

**Successful Post-Conditions**:

Customer has modified ticket for their booking.

The system has updated the information correctly.

**Failed Post-Condition:**

Customer has failed to modify a ticket for their booking because the boarding pass has already been printed.

##### Flow of Events

1. Customer selects Manage Bookings from menu
2. Customer selects a booking s/he has made.
3. Customer selects a ticket to modify
4. Customer makes changes to passenger name, seat preference, or meal preference.
5. Customer confirms and saves the modification.
6. The system updates the information.

*The system updates the manifest of the flight and meal preference list.*

1. End use-case.

##### Special Requirements

Customer can stop the modification at any step before confirmation (step 3).

##### Extension Points

None.

#### Use Case 5 – Cancel Booking (F.004)

##### Description

This use case describes how a customer can cancel an entire booking over the Internet.

##### Pre-Conditions

Customer has created a valid account.

Customer has logged into the system.

Customer’s identity has been verified.

Customer has made a booking.

The boarding pass for the booking has not been printed.

##### Post-Conditions

**Successful Post-Conditions**:

Customer has cancelled his/her booking.

Each ticket for that booking has been cancelled.

The system has updated the information correctly.

**Failed Post-Condition:**

Customer has failed to cancel his/her booking because the boarding pass has already been printed.

Some tickets for the booking remain valid.

##### Flow of Events

1. Customer selects Manage Bookings from menu
2. Customer selects a booking they have made, and clicks Cancel Booking.
3. Customer is shown a summary of the booking.
4. Customer confirms the cancellation.
5. The system updates the information.

*The system updates the manifest of the flight and issues a refund.*

1. End use-case.

##### Special Requirements

Customer can stop the cancellation at any step before confirmation (step 3).

##### Extension Points

None.

#### Use Case 6 – View and Print Boarding Passes (F.005)

##### Description

This use case describes how a customer can view and/or print his/her boarding pass for the purchased flight tickets.

##### Pre-Conditions

Customer has created a valid account.

Customer has logged into the system.

Customer’s identity has been verified.

Customer has paid for his/her flight reservation.

##### Post-Conditions

Customer has confirmed the purchase.

Customer has viewed his/her boarding pass.

Customer has printed his/her boarding pass.  
Customer has been granted any associated points for that booking.   
Customer may no longer modify or cancel his ticket or booking.

##### Flow of Events

1. Customer selects Manage Bookings from menu.
2. Customer selects a ticket s/he has purchased.
3. Customer views the boarding pass.
4. Customer prints the boarding pass.
5. The system records that boarding pass has been viewed.

*Customer will not be able to make any changes to this reservation.*

1. The system updates the customer’s loyalty points balance.

*Loyalty points are only granted if booking was paid with credit card.*

1. End use-case.

##### Special Requirements

Customer should not be able to make modifications or cancel the booking once a boarding pass has been printed.

##### Extension Points

None.

### Airline Staff and Management Use Cases

#### Use Case 1 – View Flight Manifest (Airline Staff) (F.100)

##### Description

This use case describes how airline staff can view a flight manifest.

##### Pre-Conditions

User has logged in using an account with staff-level privileges.

##### Post-Conditions

Flight manifest is shown for selected flight.

##### Flow of Events

1. User selects View Flight Manifest option from menu.
2. User is presented with a list of flight manifest available to user.
3. User clicks on one of the flights listed.
4. The system presents manifest of the selected flight.
5. End use case.

##### Special Requirements

User can go back to previous steps at any time.

##### Extension Points

None.

#### Use Case 2 – View Flight Manifest (Airline Managers) (F.101)

##### Description

This use case describes how airline managers can view a flight manifest.

##### Pre-Conditions

User has logged in using an account with manager-level privileges.

##### Post-Conditions

**Successful Post-Condition**:

Flight manifest is shown for selected flight.

**Failed Post-Condition:**

Error message is shown to the user.

##### Flow of Events

1. User selects View Flight Manifest option from menu.
2. User is presented with a list of all flight manifests.
3. User clicks on one of the flights listed.
4. The system presents manifest of the selected flight.
5. End use case.

##### Special Requirements

User can go back to previous steps at any time.

##### Extension Points

None.

#### Use Case 3 – View Flight Statistics (F.102)

##### Description

This use case describes how airline managers can view flight statistics.

##### Pre-Conditions

User has logged in using an account with manager-level privileges.

##### Post-Conditions

**Successful Post-Conditions**:

User is presented with flight statistics.

**Failed Post-Condition:**

An error message is shown to the user.

##### Flow of Events

1. User selects to view flight statistics option from the menu.
2. System retrieves data from database, “Loading…” message is shown to user if loading time is significant.
3. Statistics are shown to the user.
4. End use case.

##### Special Requirements

User can go back to previous steps at any time.

##### Extension Points

None.

#### Use Case 4 – Modify Ticket Reservation on Behalf of Customer (F.103)

##### Description

This use case describes how airline staff and management can modify ticket reservations on behalf of customers.

##### Pre-Conditions

User has logged in using an account with staff or manager-level privileges.

User has been provided with customer’s email address.

Customer has a booking with one or more tickets.

##### Post-Conditions

**Successful Post-Conditions**:

Ticket for the booking was successfully modified.

System updates related information properly.

**Failed Post-Condition:**

Modification failed since the customer has already printed the boarding pass for the booking.

##### Flow of Events

1. User selects Manage User Bookings from menu.
2. User enters customer’s email address.
3. System displays bookings associated with customer’s email address.
4. User selects from available bookings.
5. User selects ticket to modify.
6. User modifies any of passenger name, seat preference, meal preference
7. User confirms and saves the modification.
8. The system updates information in database.
9. End use case.

Alternative Flow A

1. System notifies user that no modification can be made since customer has already viewed the boarding pass.
2. End use case with failure condition.

Alternative Flow B

1. System cannot find matching bookings associated with provided customer email address.
2. Use case resumes at step 2.

##### Special Requirements

User can modify selection and/or go back to previous steps before confirmation.

##### Extension Points

None.

#### Use Case 5 – Cancel Booking on Behalf of Customer (F.105)

##### Description

This use case describes how airline staff and management can cancel a booking and all associated tickets on behalf of customers.

##### Pre-Conditions

User has logged in using an account with staff or manager-level privileges.

User has been provided with customer’s email.

Customer has one or more valid bookings.

##### Post-Conditions

**Successful Post-Conditions**:

Flight booking was successfully cancelled.

All tickets associated with the booking were cancelled.

System updates related information properly.

**Failed Post-Condition:**

Cancellation failed since the customer has already printed a boarding pass.

##### Flow of Events

1. User selects Manage User Bookings from menu.
2. User enters customer email.
3. System displays bookings associated with customer’s email address.
4. User locates the target booking and clicks Cancel Booking.
5. System displays a summary of selected booking.
6. User confirms the cancellation.
7. The system updates information in database.
8. End use case.

Alternative Flow A

1. System notifies user that cancellation cannot be made for target booking since boarding pass has been viewed.
2. End use case with failure condition.

Alternative Flow B

1. System cannot find matching bookings associated with provided customer email address.
2. End use case with failure condition.

##### Special Requirements

User can go back to previous steps before confirmation.

##### Extension Points

None.

# Architecture

## Architecture of Struts, Tools Used and Patterns Followed

During the design stage of this project, Pod Ten decided that due to the large scope of the project, it was better to spend more time on the application itself, rather than the building the infrastructure to support the application. For this reason, we decided on a web application framework that would consist of Tomcat as the web application server, Apache Struts as the web framework, and FreeMarker as the templating engine.

We decided that using an MVC framework like Struts or SpringMVC would make it difficult for us to stray from an MVC design. We considered several MVC frameworks, but we narrowed them down to Struts and SpringMVC. Both of these frameworks had a broad user base, were very reliable and had great documentation and other resources. We ended up choosing Struts because of the centralized xml file based configuration it offered, versus SpringMVC where the URL’s would have to be mapped via annotations within the Java code.

We debated between two different templating technologies- JSP and FreeMarker. JSP seemed to be more commonly used, but we realised that FreeMarker ensured a better separation between presentation and business logic. We realised that this would help us better enforce an MVC development philosophy. FreeMarker’s documentation was not very extensive, but it was very easy to follow. There was also some prior experience within the team, so we decided to go with that over JSP.

To manage our project structure, we decided to use Maven dependency management tool. This enabled us to add external jars and dependencies without ever having to worry about version numbers. It also supported integration with JUnit, which was to be our automated test framework.

Figure 18 below explains the architecture of the AirPlan application. Apache Tomcat passes the HTTP request to Struts’ action mapper. Struts uses its central xml configuration file to map the HTTP request to a Java action. It passes parameters from the GET or POST request and sets them as the action’s class parameters using setters.

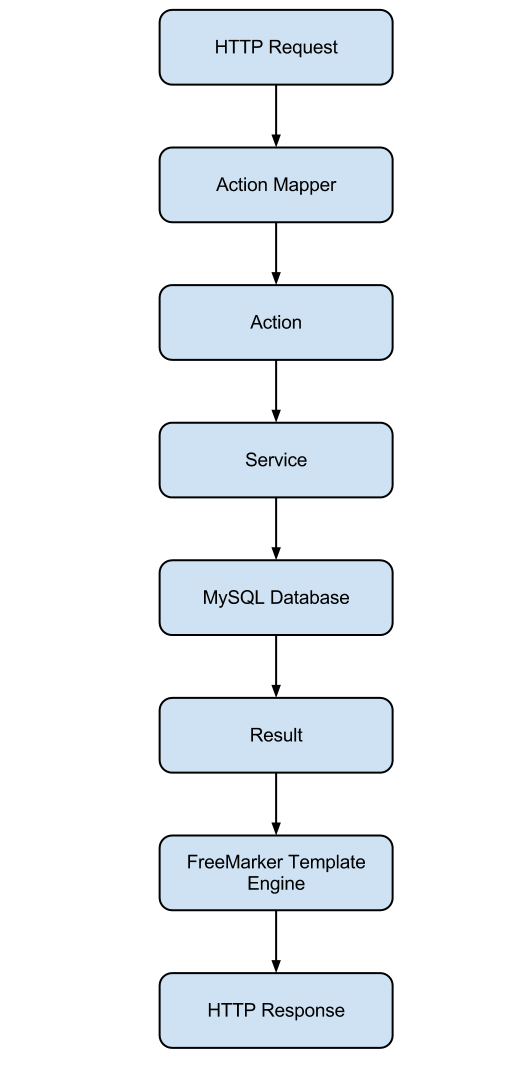


Figure 37: Struts Architecture

In our architecture, the actions do not contain any business logic. Instead, each action calls one or more service layer functions. Each service layer function contains some business logic for that use case. This includes all related database queries and updates. When the service layer is done, it returns whatever data is to be displayed to the action. The action then passes this data to the templating engine (FreeMarker) to create the HTTP response.

This allows us to re-use our service layer code as much as possible. For this reason, there is not a one to one mapping between actions and service layer functions. With this, we have fewer lines of code, and lesser code to maintain.

This is also described in the package diagram below (Figure 38). As seen from it, each action uses service as well as domain classes to perform its functions. Here, each domain class represents a table in the database. Hence their fields map directly to the fields in the database.

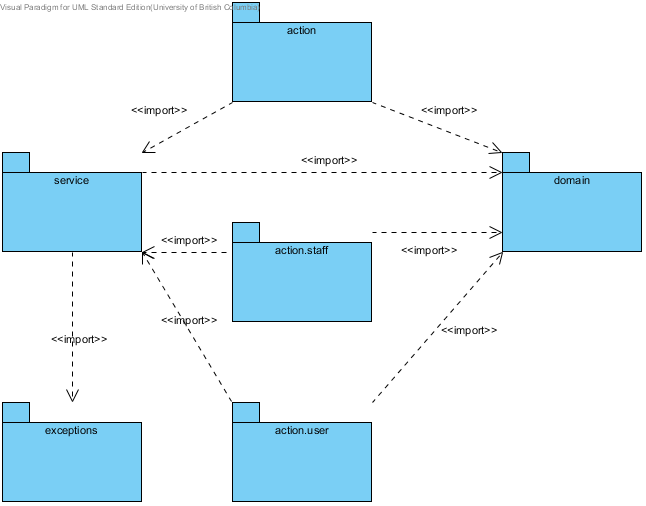
****

Figure 38: Package Diagram

The service layer class diagram below (Figure 39) shows the relationships between different services, as well as their relationships with domain classes.

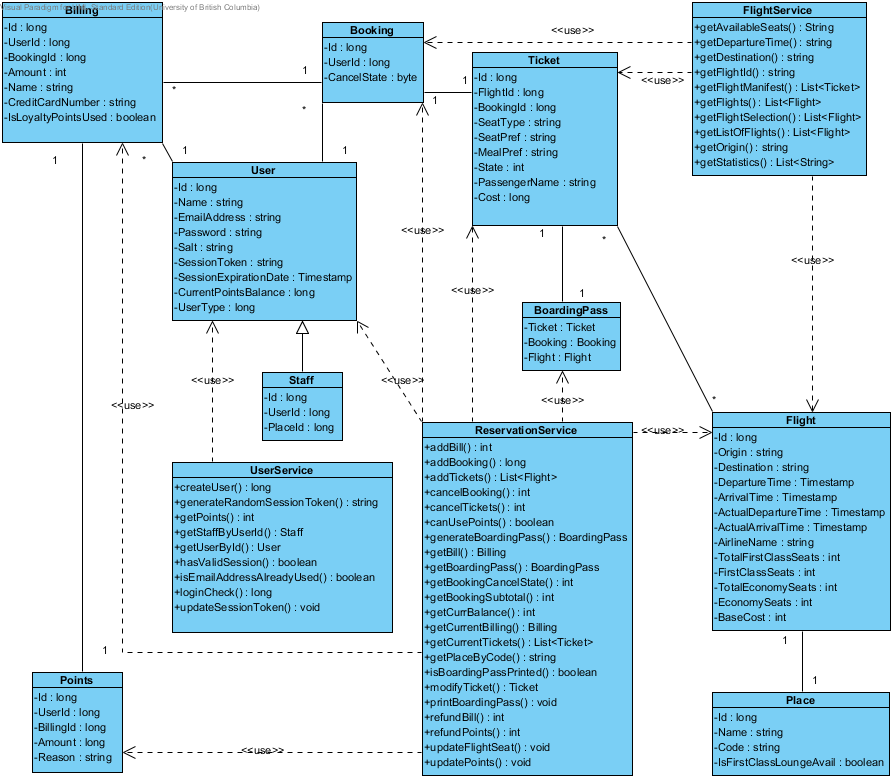
****

Figure 39: Service Layer Class Diagram

The action layer class diagram shows all the different actions that are used in this application, and how they all inherit from BaseAction. BaseAction contains some common functionality that all other actions use, hence this hierarchy is essential.

As it can be seen, following this architecture allowed us to develop our system using the MVC pattern. This architecture ensured that the representation of data was always done within our domain classes (the model), the business logic was kept inside the service layer (the controller), and the presentation logic was kept in the action layer (the view). Once members of the team got used to this working within this framework, it became very easy to expand to new features and to maintain existing ones.

Another architectural pattern that we followed was the Singleton pattern. This was used to initiate and maintain a database connection to the MySQL database. This was done to ensure that only one connection was ever created per session, so that the overall system operated more efficiently.

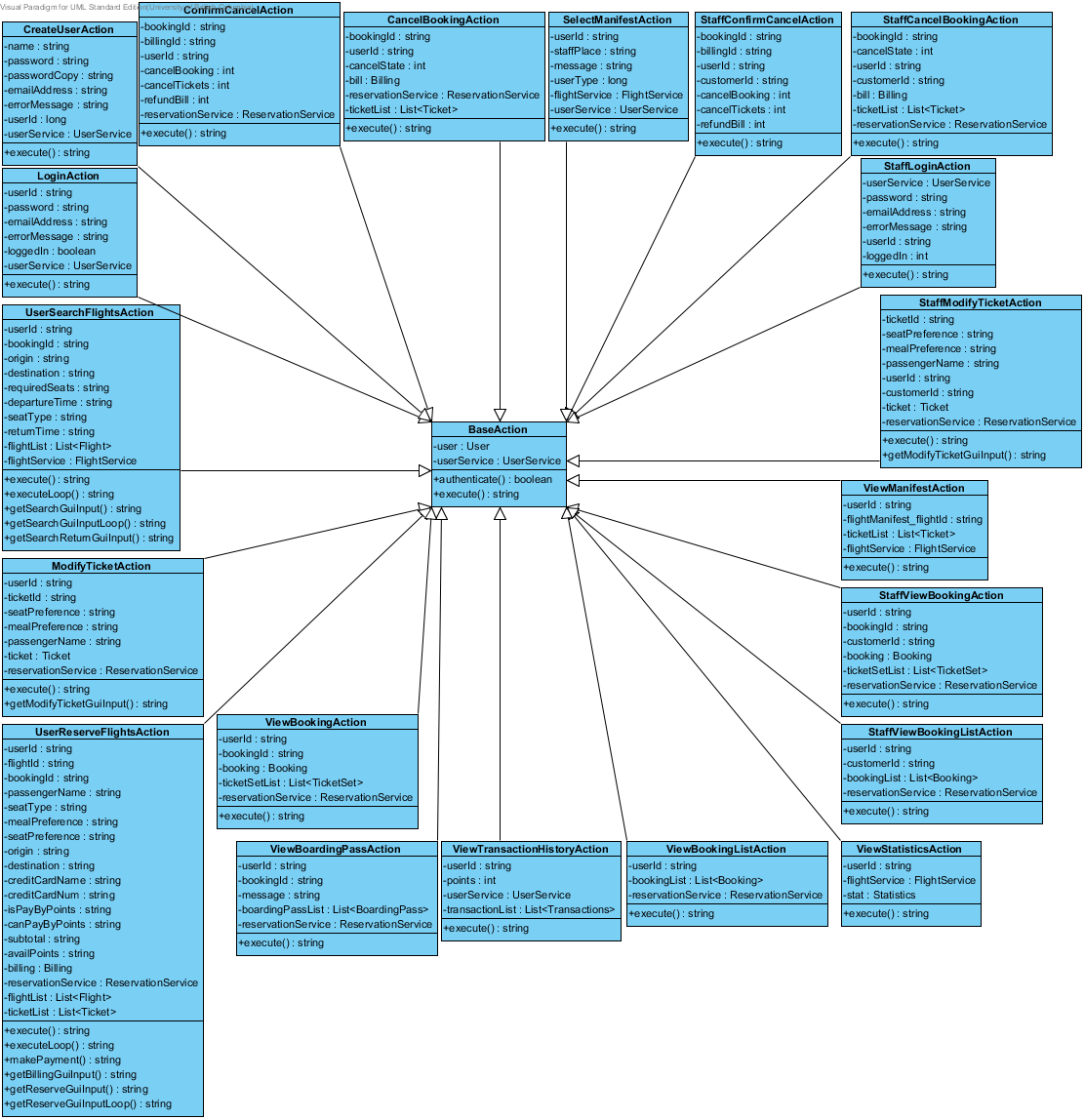
****

Figure 40: Action Layer Class Diagram

## Sequence Diagrams

### Search

The following sequence diagram describes how a customer can search for flights over the Internet.

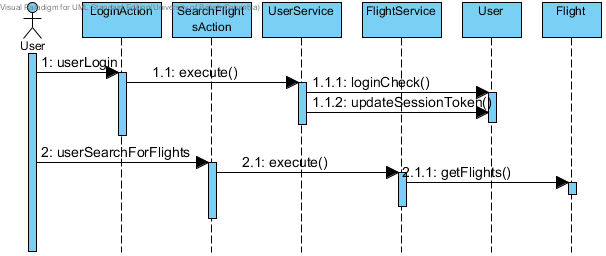


Figure 41: Search Flights Sequence Diagram

### Reserve

The following sequence diagram describes how a customer can reserve flights using the AirPlan reservation system.

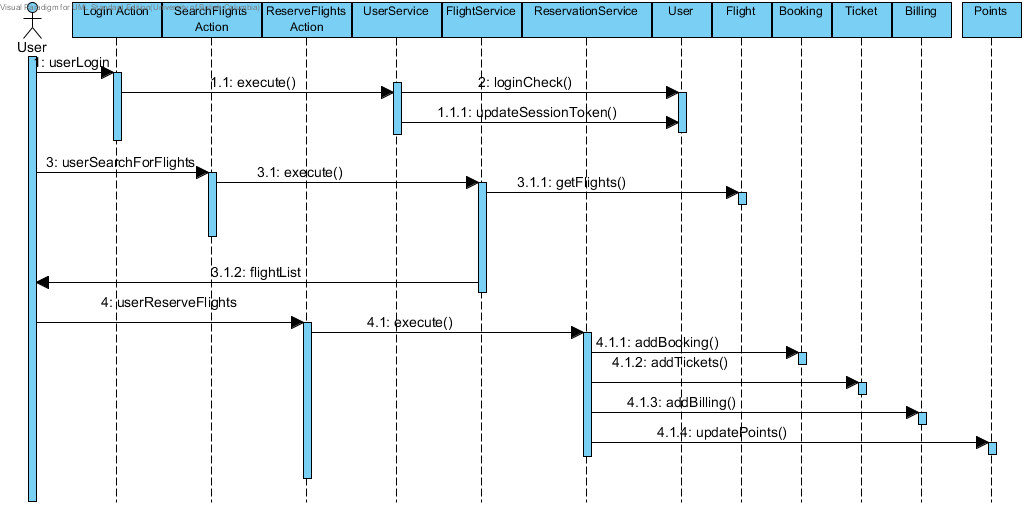


Figure 42: Reserve Flight Sequence Diagram

### View Transactions

The following sequence diagram describes how a customer can view all their past transactions and check their loyalty points usage and history.

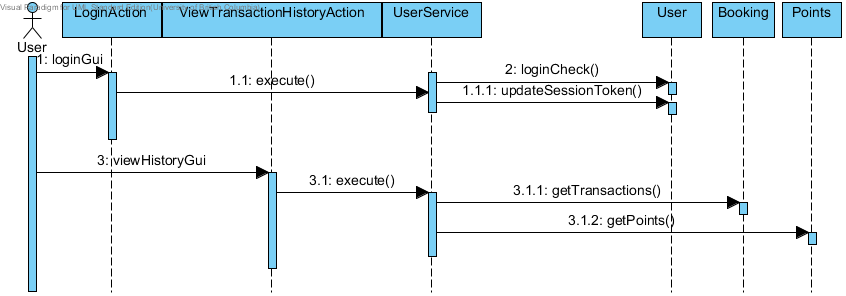


Figure 43: Manage Payment Sequence Diagram

### Print Boarding Pass

The following sequence diagram describes how a customer can view and/or print their boarding pass for the purchased flight tickets.

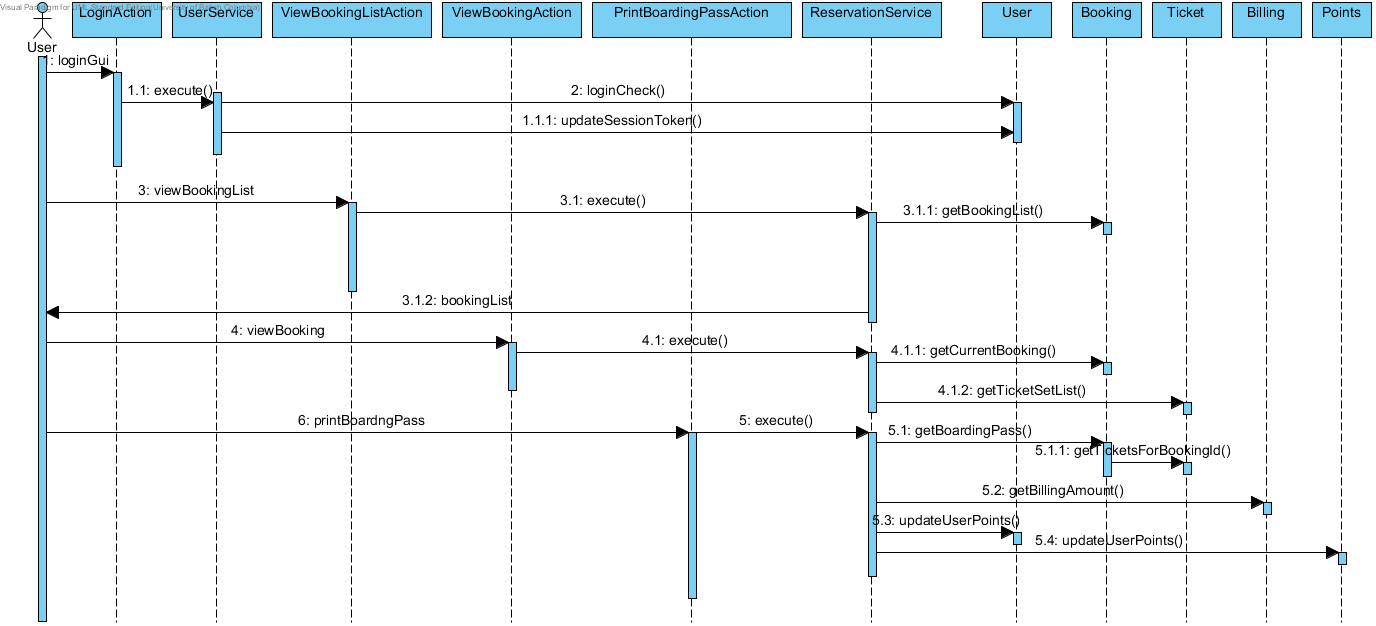


Figure 44: Print Boarding Pass Sequence Diagram

### Modify Reservation

The following sequence diagram describes how a customer can modify their ticket reservation over the Internet.

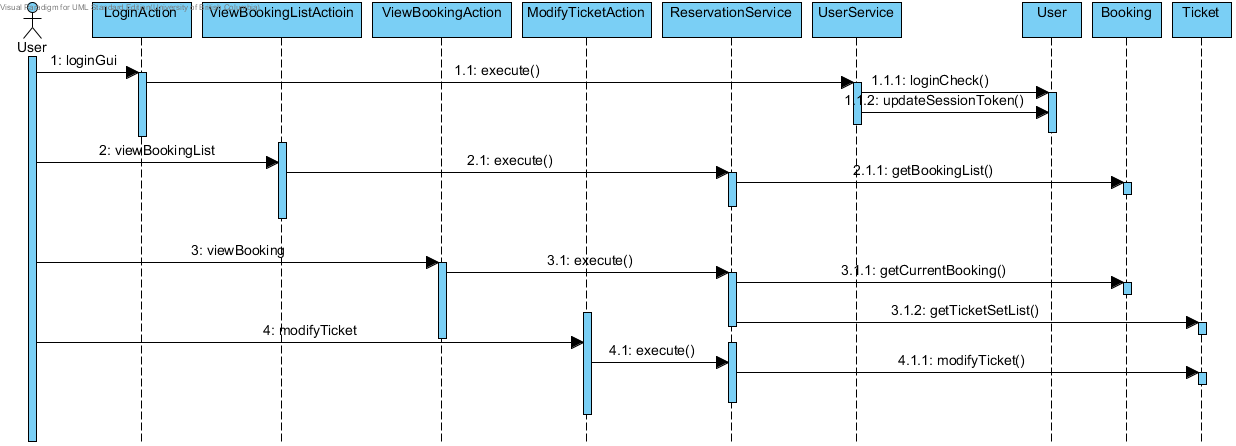


Figure 45: Modify Reservation Sequence Diagram

### Cancel Reservation

The following sequence diagram describes how a customer can cancel their booking over the Internet.

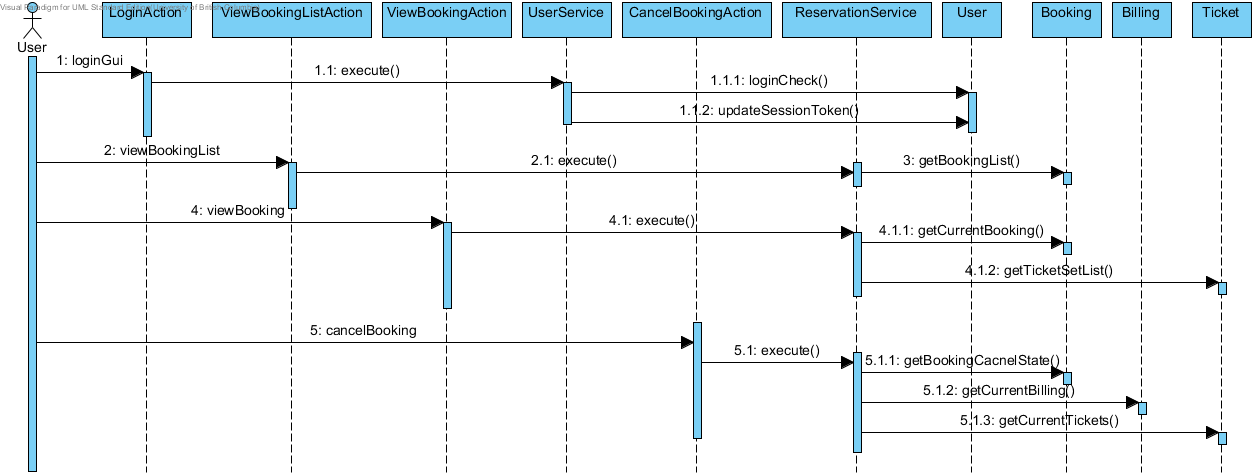


Figure 46: Cancel Reservation Sequence Diagram

### View Flight Manifest (Staff)

The following sequence diagram describes how airline staff can view a flight manifest.

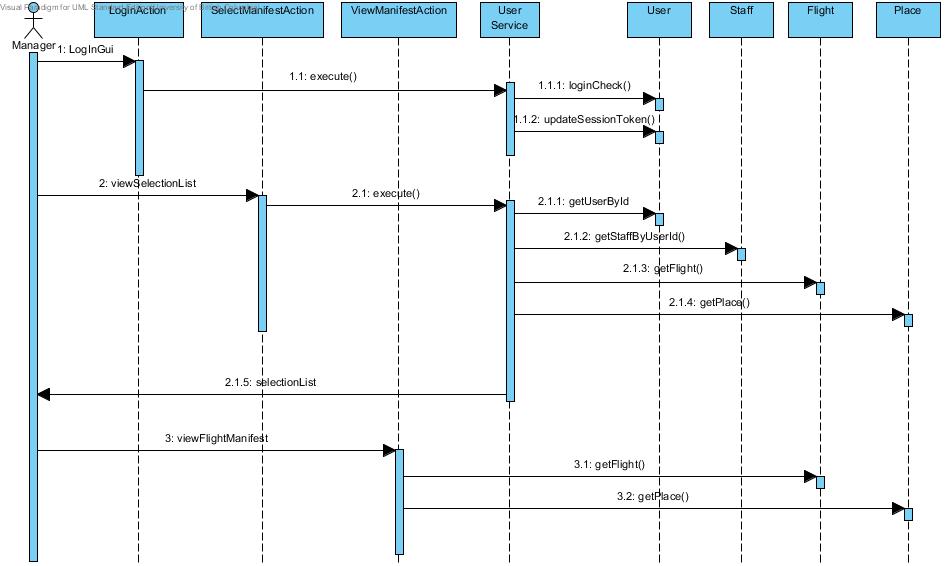


Figure 47: View Flight Manifest Sequence Diagram

### View Flight Manifest (Management)

The following sequence diagram describes how airline managers can view a flight manifest.

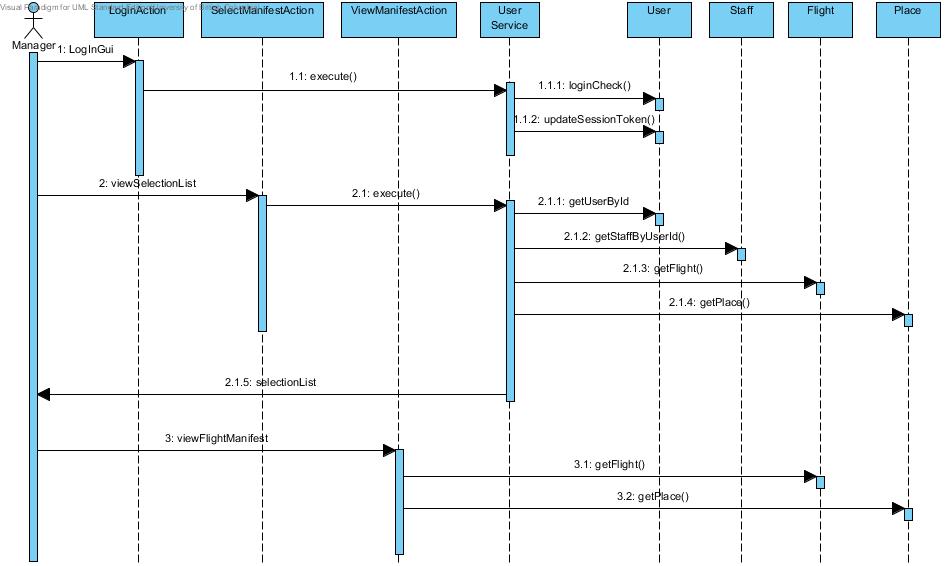


Figure 48: View Flight Manifest Sequence Diagram

### View Flight Statistics

The following sequence diagram describes how airline managers can view flight statistics.

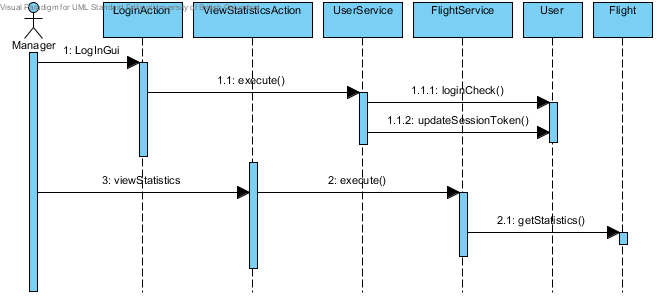


Figure 49: View Flight Statistics Sequence Diagram

### Modify Reservation on Behalf of Customers

The following sequence diagram describes how airline staffs/managers can modify ticket reservations on behalf of customers.

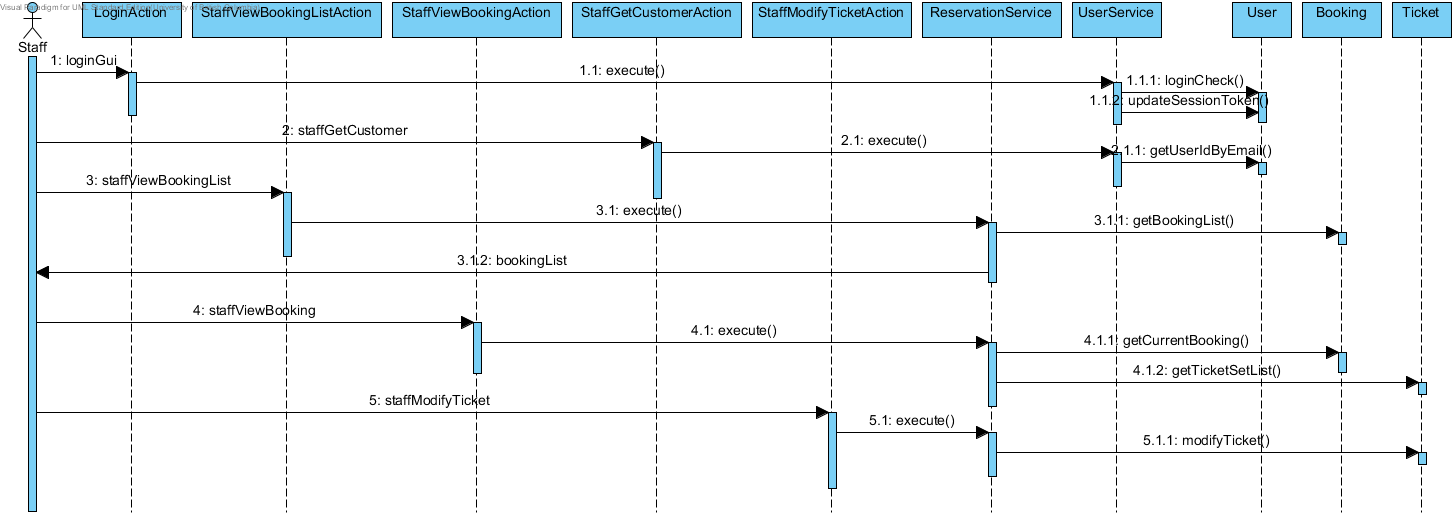


Figure 50: Modify Reservation on Behalf of Customers

### Cancel Reservation on Behalf of Customers

The following sequence diagram describes how airline staffs/managers can cancel bookings on behalf of customers.

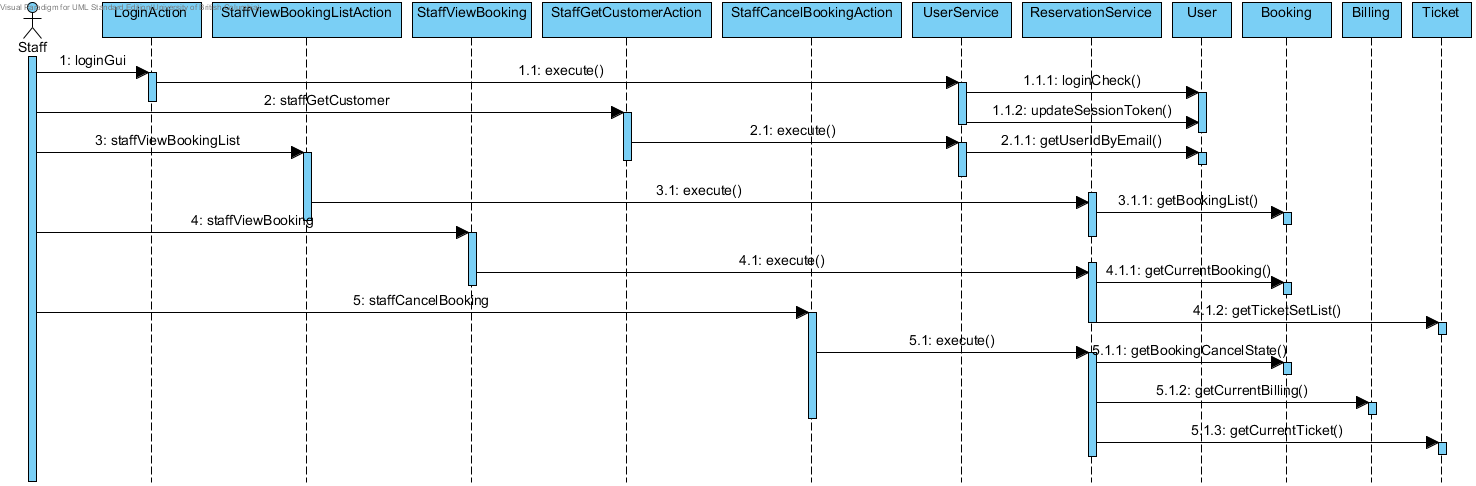


Figure 51: Cancel Reservation on Behalf of Customers

## Deployment Architecture

The AirPlan flight management and reservation system will be deployed on a server rented from a hosting service provider such as Amazon AWS. AirPlan’s database will be deployed on a dedicated server so that it can be scaled independently of the application server.

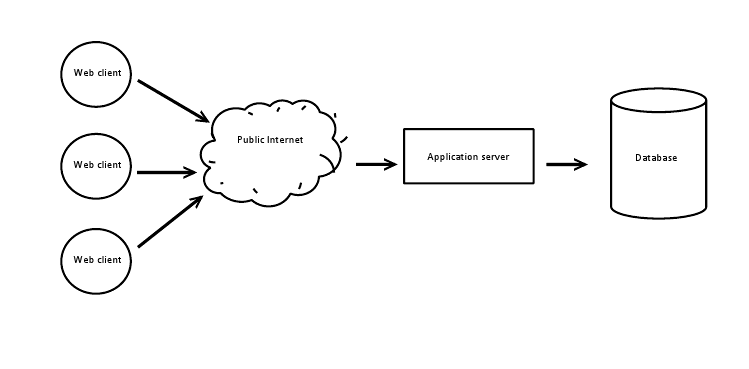


Figure 52: Deployment Diagram

# Interesting Design Aspects

## User Authentication and Security

Even though this course did not focus on security, we took several steps to protect ourselves and our users from threat agents. To ensure user security, we tried to ensure that the user was authenticated with each action. To enable this, when a user logs in, a session token and expiration date is assigned to the user. The session token is passed to the server with each request, and the user is authenticated with this token.

We also do not store passwords in plain text in the database; instead we randomly generate a salt for each password, and store the salted password in the database. We also made sure that only prepared statements are used to query the database, and user input is sanitized before being used. This will help protect AirPlan from SQL injection and cross site scripting attacks. We have not tried to be exhaustive in our efforts to secure the application, but we have tried to provide a minimum level of security to our users.

## Front-end engineering

We understood from very early on that one of the key aspects of this project is the user interface. In fact, it is known in the industry that 70-80% of performance gains for websites come from better front-end engineering. For this reason, everyone on the team was very careful when it came to front-end design, and we even had one team member focus solely on front-end engineering and front-end performance enhancements. We realised that although we wanted our web-app to run well on every browser, due to time constraints, it was not possible to test and develop for every browser. To strike a balance, we decided to use graded browser support. We created three grades of browsers with varying amounts of support. For example, the webapp would be fully supported and tested on Chrome. We would provide only partial support for users of Firefox, Safari and Opera, and we would not support any version of Internet Explorer. Table 8 below explains this in detail.

|  |  |  |
| --- | --- | --- |
| Grade | Amount of Support | Browsers |
| Grade-A | Site will be fully tested on these browsers. All JavaScript and CSS will be supported. | Chrome, Firefox |
| Grade-B | Site will be partially supported on these browsers. JavaScript will not be supported. | Safari, Opera |
| Blacklisted | No quality assurance will be done for these browsers. | Internet Explorer, Dolphin, Android Stock Browser etc. |

Table 8: Graded Browser Support

We took a number of steps to improve our front-end performance and responsiveness. One of our guiding philosophies was to reduce the overall number of HTTP requests. Latency is one of the biggest factors affecting the performance and responsiveness of a website, and by reducing the number of HTTP requests, you are directly reducing latency.

We also ensured that all our CSS would be located in one central file (airplan\_css.css). This way, when a user first arrives at our site, he will take a bigger hit because he will download a larger file where sections of the CSS are unused. However, for every subsequent request, the browser will have cached the CSS file so it will not have to request it again. On the other hand, if we had decided to use different CSS files for each page, the browser would have had to make separate HTTP requests for each CSS file. Due to the higher amount of latency, this would definitely result in a slower experience overall. Similarly, we have tried to follow the same model for our JavaScript files; nearly all of our JavaScript code is contained within two files. We also held back on the usage of icons and images in our application to further reduce the number of HTTP requests a client browser would have to make on loading any page.

One of the other techniques we used to optimize our website was to place the CSS imports at the top of each file, and the JavaScript imports at the bottom of each file. Placing the CSS import statement at the top of an HTML page ensures that the page is being styled as it is being loaded. This will ensure that the user does not see a flash of an un-styled page before viewing the fully styled page. For pages where JavaScript import statements are located at the top of markup, some browsers wait for the JavaScript file to be downloaded before rendering any UI elements. This is done so that the browser avoids missing any important DOM events that the JavaScript might take action on. Our application does not benefit from this situation, however, so we decided it was best to leave the JavaScript import at the end to avoid potential slowdowns.

We also tried not to use inline CSS as much as possible. By mainly using styling from our CSS files, we increase the ease of maintenance and debugging, and avoid having to deal with CSS conflicts that may arise due to its hierarchical nature.

While writing JavaScript we tried to be as unobtrusive as possible. We mostly avoided writing JavaScript in the HTML documents, maintained a philosophy of not trusting JavaScript checks (and varying browser support) and avoided the use of global variables. This ensured ease in JavaScript maintenance, something that is rare as your code-base grows.

Apart from greatly increasing performance, following these front-end engineering best practices contributed to higher quality of code across our web-app. This increased the maintainability of our application and will decrease the ramp up time for any new developer joining the team.

## Other Features

Pod Ten has taken extensive care to ensure that any modification to the database via the AirPlan application is logged accurately. Each log entry will include user id, timestamp and a verbal explanation of the action. Logs will also include prior and updated values where appropriate. This is to ensure that the logs are able to support the investigation of any bugs in a production environment.

We have tried to make effective use of exceptions in our service layer. For example, if an exception is thrown in a service, a meaningful exception is propagated to the action. This enables us to display a meaningful message to end users. Exceptions are also logged appropriately, to support the investigation of bugs.

We also included a dateModified timestamp field in every table in our database. This field is updated every time a row is updated. This is also to help investigate production issues by tracking the timeline of various events in the database.

# Appendix A: Test Cases and Test Scripts

All test cases have been included in this appendix. The automated test scripts have also been included here, but the Java code used to write these test scripts is included in Appendix D with the rest of the code.

## Customer Use Cases

### Search Flight

|  |  |
| --- | --- |
| Requirement ID | F.000 |
| Test Case ID | TCF.000.00 |
| Test Case Name | Manual Test: Correct Search |
| Test Steps and Test Data | Go to search page. Enter criteria for origin, destination and date of flight.  View results. |
| Expected Result | Returns a list of all flights operating between the two locations on that date are returned. |
| Pass Fail Criteria | Pass if all expected flights are returned.  Fail if a valid flight is missing (not returned).  Fail if an invalid flight (not matching the parameters) is returned. |

|  |  |
| --- | --- |
| Requirement ID | F.000 |
| Test Case ID | TCF.000.01 |
| Test Case Name | Manual Test: Incomplete Search |
| Test Steps and Test Data | Go to search page. Enter a search with one of the following fields empty: origin, destination, or date. View result. |
| Expected Result | Return an error message. |
| Pass Fail Criteria | Pass if the error message is displayed and the search does not execute.  Fail if the search executes and the user proceeds to the next step. |

|  |  |
| --- | --- |
| Requirement ID | F.000 |
| Test Case ID | TCF.000.02 |
| Test Case Name | Manual Test: Same Locations |
| Test Steps and Test Data | Go to search page. Search for flights using the same origin and destination.  View result. |
| Expected Result | An error message is displayed and a location must be changed before the search continues |
| Pass Fail Criteria | Pass if the search does not run.  Fail if the search runs and the user proceeds to the next step. |

|  |  |
| --- | --- |
| Requirement ID | F.000 |
| Test Case ID | TCF.000.03 |
| Test Case Name | Manual Test: One Way Flight Search |
| Test Steps and Test Data | Go to search page.  Click on the option to search for one way flights. View updates on page. |
| Expected Result | Options to filter for return flights are unavailable. |
| Pass Fail Criteria | Pass if the search page does not include a return date.  Fail if the search page prompts for a return date. |

|  |  |
| --- | --- |
| Requirement ID | F.000 |
| Test Case ID | TCF.000.04 |
| Test Case Name | Manual Test: Round-trip Flight With No Return Date |
| Test Steps and Test Data | Go to search page. Click on the option to search for a round trip flight. Search using a return date blank. View result. |
| Expected Result | An error message is displayed and the search does not continue |
| Pass Fail Criteria | Pass if an error message is returned and the search does not continue.  Fail if the search proceeds without failure. |

|  |  |
| --- | --- |
| Requirement ID | F.000 |
| Test Case ID | TCF.000.05 |
| Test Case Name | Manual Test: Round-trip Flight Search |
| Test Steps and Test Data | Go to search page. Search for a round-trip flight. |
| Expected Result | Return two sets of flights, one list of departing flights and one list of returning flights. |
| Pass Fail Criteria | Pass if two lists of flights are returned.  Fail if the search returns less or more than two lists. |

### Make a Reservation

|  |  |
| --- | --- |
| Requirement ID | F.001 |
| Test Case ID | TCF.001.00 |
| Test Case Name | Manual Test: Set Reservation |
| Test Steps and Test Data | Go to search page. Search for flights, and select a flight from the flights that are returned.  Select flight options. |
| Expected Result | User is sent to billing page. |
| Pass Fail Criteria | Assert that billing page is loaded. Pass if amounts billed for are correct.  Fail if billing page not returned or wrong billing amounts returned. |

|  |  |
| --- | --- |
| Requirement ID | F.001 |
| Test Case ID | TCF.001.01 |
| Test Case Name | Manual Test: Billing |
| Test Steps and Test Data | Go to search page. Search for flights, and select a flight from the flights that are returned.  Select flight options. Enter billing information and number of points to use. |
| Expected Result | Reservation summary is displayed. |
| Pass Fail Criteria | Assert that reservation summary is loaded and data on this summary is correct. Fail otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F.001 |
| Test Case ID | TCF.001.02 |
| Test Case Name | Manual Test: Enter invalid billing information. |
| Test Steps and Test Data | Go to search page. Search for flights, and select a flight from the flights that are returned.  Select flight options. Enter invalid billing information, or try to use more points than are available. |
| Expected Result | An error message is displayed that the billing information is invalid or that account has less points than specified. |
| Pass Fail Criteria | Assert that error message is returned.  Fail if user is allowed to proceed without error message. |

### View Transaction History

|  |  |
| --- | --- |
| Requirement ID | F.002 |
| Test Case ID | TCF.002.00 |
| Test Case Name | Manual Test: View transactions for billing paid with credit card with boarding pass not yet printed. |
| Test Steps and Test Data | Create a booking for the user but do not print the boarding pass.  Go to view transaction history page. |
| Expected Result | Display all transactions including recently created booking. The recently created transaction should note that points will be awarded once boarding pass is printed. |
| Pass / Fail Criteria | Pass if correct information is displayed.  Fail if information not displayed or points have been awarded |

|  |  |
| --- | --- |
| Requirement ID | F.002 |
| Test Case ID | TCF.002.01 |
| Test Case Name | Manual Test: View transactions for billing paid with credit card with boarding pass printed. |
| Test Steps and Test Data | Create a booking for the user and print the boarding pass.  Go to view transaction history page. |
| Expected Result | Display all transactions including recently created booking. The recently created transaction should have points awarded equal to the cost of the flight. |
| Pass / Fail Criteria | Pass if correct information is displayed.  Fail if information not displayed or points have not been awarded. |

|  |  |
| --- | --- |
| Requirement ID | F.002 |
| Test Case ID | TCF.002.02 |
| Test Case Name | Manual Test: View transactions for billing paid with points. |
| Test Steps and Test Data | Create a booking for the user paid for with points.  Go to view transaction history page. |
| Expected Result | Display all transactions including recently created booking. The recently created transaction should note it is paid for with points and thus no points will be awarded for the transaction. |
| Pass / Fail Criteria | Pass if correct information is displayed.  Fail if correct information not displayed. |

### Modify Reservation

|  |  |
| --- | --- |
| Requirement ID | F.003 |
| Test Case ID | TCF.003.00 |
| Test Case Name | Manual Test: View list of reservations for future flights |
| Test Steps and Test Data | On the homepage, select on “Manage” button. Click on “View Flight Booking” link. |
| Expected Result | A list containing all bookings made by customer for upcoming flights is returned. |
| Pass Fail Criteria | Pass if all bookings made by user are listed. Test if any of the bookings are missing. Test if any booking not made by user is incorrectly listed. |

|  |  |
| --- | --- |
| Requirement ID | F.003 |
| Test Case ID | TCF.003.01 |
| Test Case Name | Manual Test: View booking for an upcoming flight |
| Test Steps and Test Data | On the homepage, select on “Manage” button. Click on “View Flight Bookings” link. Select one of the bookings listed. |
| Expected Result | Information associated with the selected bookings is listed correctly. Departure time & location, Arrival time & location, seat location, flight class. |
| Pass Fail Criteria | Pass if all relevant information is present and listed correctly. Test if some information missing or incorrect. |

|  |  |
| --- | --- |
| Requirement ID | F.003 |
| Test Case ID | TCF.003.02 |
| Test Case Name | Manual Test: Modify seat selection for an upcoming flight |
| Test Steps and Test Data | On the homepage, select on “Manage” button. Click on “View Flight Bookings” link. Select one of bookings listed which user hasn’t printed boarding pass. Click on “Modify Tickets” button. Change seat preference to one of the options and press save button. After saving, view the booking again to make sure the changes were saved correctly. Click modify button again, make sure that the initial seat selection is now free (i.e. can be selected by other user). |
| Expected Result | Changes in seat selection are saved properly. |
| Pass Fail Criteria | Pass if user able to change seat selection and changes are successfully recorded in the ticket table in the database.  Ensure that available seating space on flight is correct after transaction.  Fail otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F.003 |
| Test Case ID | TCF.003.03 |
| Test Case Name | Manual Test: Verify cannot modify reservation when boarding pass has been printed. |
| Test Steps and Test Data | On the homepage, select on “Manage” button. Click on “View Flight Bookings” link. Select one of the bookings listed which the user has already printed boarding pass for. Verify that the “Modify Ticket” button is grayed out. |
| Expected Result | User is unable to modify booking since boarding pass has been printed. |
| Pass Fail Criteria | Pass if buttons for modify booking is grayed out. |

### Cancel Reservation

|  |  |
| --- | --- |
| Requirement ID | F.004 |
| Test Case ID | TCF.004.00 |
| Test Case Name | Manual Test: View list of reservations for future flights |
| Test Steps and Test Data | On the homepage, select on “Manage” button. Click on “View Flight Booking” link. |
| Expected Result | A list containing all bookings made by customer for upcoming flights is returned. |
| Pass Fail Criteria | Pass if all bookings made by user are listed. Test if any of the bookings are missing. Test if any booking not made by user is incorrectly listed. |

|  |  |
| --- | --- |
| Requirement ID | F.004 |
| Test Case ID | TCF.004.01 |
| Test Case Name | Manual Test: View booking for an upcoming flight |
| Test Steps and Test Data | On the homepage, select on “Manage” button. Click on “View Flight Bookings” link. Select one of the bookings listed. |
| Expected Result | Information associated with the selected bookings is listed correctly. Departure time & location, Arrival time & location, seat location, flight class. |
| Pass Fail Criteria | Pass if all relevant information is present and listed correctly. Test if some information missing or incorrect. |

|  |  |
| --- | --- |
| Requirement ID | F.004 |
| Test Case ID | TCF.004.02 |
| Test Case Name | Manual Test: Cancel booking for an upcoming flight |
| Test Steps and Test Data | On the homepage, select on “Manage” button. Click on “View Flight Bookings” link. Select one of bookings listed which user hasn’t printed boarding pass. Click on “Cancel Booking” button, press Yes when asked to confirm. After cancelling, view the booking list to make sure the reservation is marked as cancelled. |
| Expected Result | Booking is cancelled successfully and money paid by user is refunded (no points). |
| Pass Fail Criteria | Pass if user able to cancel booking, and changes are successfully recorded in the booking and billing table in the database. Fail otherwise. Test if system does not issue refund. |

|  |  |
| --- | --- |
| Requirement ID | F.004 |
| Test Case ID | TCF.004.03 |
| Test Case Name | Manual Test: Verify cannot cancel reservation when boarding pass has been printed. |
| Test Steps and Test Data | On the homepage, select on “Manage” button. Click on “View Flight Bookings” link. Select one of the bookings listed which the user has already printed boarding pass for. Verify that the “Cancel Booking” button is grayed out. |
| Expected Result | User is unable to cancel booking since boarding pass has been printed. |
| Pass Fail Criteria | Pass if buttons for cancel booking is grayed out. |

### View and Print Boarding Pass

|  |  |
| --- | --- |
| Requirement ID | F.004 |
| Test Case ID | TCF.004.00 |
| Test Case Name | Manual Test: Testing view boarding pass with valid booking id. |
| Test Steps and Test Data | Click on Manage Bookings. Select a booking from list of bookings to view booking details. Click on print boarding pass(es) button. Ensure boarding pass(es) are printed correctly. |
| Expected Result | Associated boarding passes are printed correctly. |
| Pass Fail Criteria | Fail if any incorrect boarding pass is printed. Fail if any associated boarding passes are not printed.  Fail if user is still allowed to cancel flight. Pass otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F.004 |
| Test Case ID | TCF.004.01 |
| Test Case Name | Manual Test: Testing view boarding pass with already confirmed booking id. |
| Test Steps and Test Data | Click on Manage Bookings. Select a booking that is already confirmed from list of bookings to view details. Click on print boarding pass(es) button. Ensure boarding pass(es) are printed correctly. |
| Expected Result | Associated boarding passes are printed correctly. |
| Pass Fail Criteria | Fail if any incorrect boarding pass is printed.  Fail if any associated boarding passes are not printed. Fail if user is awarded points for printing boarding pass.  Fail if user is allowed to cancel flight. Pass otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F.004 |
| Test Case ID | TCF.004.02 |
| Test Case Name | Manual Test: Testing print boarding pass of booking that has not been confirmed yet. |
| Test Steps and Test Data | Click on Manage Bookings. Select a booking that is not yet already confirmed from list of bookings to view details. Click on print boarding pass(es) button. Ensure boarding pass(es) are printed correctly. |
| Expected Result | Associated boarding passes are printed correctly. |
| Pass Fail Criteria | Fail if any incorrect boarding pass is printed.  Fail if any associated boarding passes are not printed. Fail if user is not awarded any required points.  Fail if user is allowed to cancel flight. Pass otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F.004 |
| Test Case ID | TCF.004.06 |
| Test Case Name | Unit Test: Testing get boarding pass with a valid booking id |
| Test Steps and Test Data | Query a boarding pass with a valid input booking id |
| Expected Result | Return boarding pass information for the corresponding booking id |
| Pass Fail Criteria | Assert that the booking id is valid, and the booking has not been confirmed yet.  Fail otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F.004 |
| Test Case ID | TCF.004.07 |
| Test Case Name | Unit Test: Testing get boarding pass with an already confirmed booking id. |
| Test Steps and Test Data | Query an already-confirmed boarding pass with a valid input booking id |
| Expected Result | Return error code already confirmed |
| Pass Fail Criteria | Assert that booking id is valid, and that the booking has already been confirmed.  Pass if correct error code is returned. Fail otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F.004 |
| Test Case ID | TCF.004.08 |
| Test Case Name | Unit Test: Testing get boarding pass with an invalid booking id |
| Test Steps and Test Data | Query a boarding pass with an invalid input id |
| Expected Result | Return error code invalid booking id |
| Pass Fail Criteria | Assert that booking id is invalid.  Pass if correct error code is returned.  Fail otherwise. |

## Staff and Management Use Cases

### View Flight Manifest (Airline Staff)

|  |  |
| --- | --- |
| Requirement ID | F.100 |
| Test Case ID | TCF.100.00 |
| Test Case Name | Manual Test: Getting a list of flight manifests (staff level) |
| Test Steps and Test Data | Go to the flight manifest page by selecting "View Flight Manifests" from menu. |
| Expected Result | List of flight manifests available for viewing is retrieved and displayed. |
| Pass / Fail Criteria | Pass if list of flight manifests is correctly displayed. Fail if staff can view flight manifests they should not have access to. Fail if staff cannot view flight manifests they should have access to. Fail if there is insufficient error management (e.g. when list is unavailable / inaccessible). |

|  |  |
| --- | --- |
| Requirement ID | F.100 |
| Test Case ID | TCF.100.01 |
| Test Case Name | Manual Test: Viewing a specific flight manifest |
| Test Steps and Test Data | Select "View Flight Manifests" from menu. Select a flight from the list of flight manifests. |
| Expected Result | Correct flight manifest is displayed for the selected flight. |
| Pass / Fail Criteria | Pass if correct flight manifest is displayed. Fail if staff cannot view flight manifest they should have access to.  Fail is there is insufficient error management (e.g. when manifest is unavailable / inaccessible). |

|  |  |
| --- | --- |
| Requirement ID | F.100 |
| Test Case ID | TCF.100.02 |
| Test Case Name | Manual Test: Return to list and select another flight manifest |
| Test Steps and Test Data | Select "View Flight Manifests" from menu. Select a flight from list of flight manifests. Click "Return to List" after selecting a flight. Select another flight from list. |
| Expected Result | Correct flight manifest is displayed for the selected flight. |
| Pass / Fail Criteria | Fail if staff can view flight manifests they should not have access to after returning to list. Fail if staff cannot view flight manifests they should have access to after returning to list. Fail if the second manifest displayed is an incorrect one. Fail if there is insufficient error management.  Pass otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F.100 |
| Test Case ID | TCF.100.03 |
| Test Case Name | Manual Test: Viewing details of a flight manifest |
| Test Steps and Test Data | Select "View Flight Manifests" from menu. Select a flight from list of flight manifests. Select a specific seat on the manifest. |
| Expected Result | Correct passenger's name is displayed, along with their meal plan, and seating class. |
| Pass / Fail Criteria | Fail if incorrect seat code displayed. Fail if incorrect passenger name displayed. Fail if incorrect meal plan for the passenger. Fail if incorrect seating class for the passenger.  Pass otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F.100 |
| Test Case ID | TCF.100.04 |
| Test Case Name | Manual Test: Return to manifest and view another seat’s details |
| Test Steps and Test Data | Select "View Flight Manifests" from menu. Select a flight from list of flight manifests. Select a specific seat on the manifest. Click "Return to Manifest" after selecting a seat. Select another seat from manifest. |
| Expected Result | Correct passenger's name is displayed, along with their meal plan, and seating class. |
| Pass / Fail Criteria | Fail if incorrect manifest is returned.  Fail if incorrect seat code displayed. Fail if incorrect passenger name displayed. Fail if incorrect meal plan for the passenger. Fail if incorrect seating class for the passenger.  Pass otherwise. |

### View Flight Manifest (Airline Management)

|  |  |
| --- | --- |
| Requirement ID | F.101 |
| Test Case ID | TCF.101.00 |
| Test Case Name | Manual Test: Getting a list of flight manifests (manager level) |
| Test Steps and Test Data | Select "View Flight Manifests". |
| Expected Result | Search/Filter Flights page displayed. |
| Pass / Fail Criteria | Pass if correct search page displayed. Pass if all fields on page are correctly labelled and populated. Fail if there is insufficient error management. |

|  |  |
| --- | --- |
| Requirement ID | F. 101 |
| Test Case ID | TCF. 101.01 |
| Test Case Name | Manual Test: Submit search form for flight manifests |
| Test Steps and Test Data | Select "View Flight Manifests" from menu. Fill out criteria fields and submit form. |
| Expected Result | Correct flight manifest list displayed, with applied criteria and filters. |
| Pass / Fail Criteria | Fail if date fields do not restrict input to correct input (e.g. dd/mm/yy). Fail if time fields do not restrict input to correct input (e.g. hh:mm:ss). Fail if input illogical (e.g. end date is comes before start date). Fail if returned list does not follow submitted criteria/filters. Fail if there is insufficient error management (e.g. no error message when no flights match criteria). |

|  |  |
| --- | --- |
| Requirement ID | F. 101 |
| Test Case ID | TCF. 101.02 |
| Test Case Name | Manual Test: Return to filter page and submit new criteria/filters. |
| Test Steps and Test Data | Select "View Flight Manifests" from menu. Fill out criteria fields and submit form. Click "Return to Filter page" after reaching list of flight manifests. Submit form with new criteria (test each field separately). |
| Expected Result | Correct flight manifest list displayed, with applied criteria and filters. |
| Pass / Fail Criteria | Fail if date fields do not restrict input to correct input (e.g. dd/mm/yy). Fail if time fields do not restrict input to correct input (e.g. hh:mm:ss). Fail if input is illogical (e.g. end date is comes before start date). Fail if filter page does not preserve / follow submitted criteria / filters when returning from manifest list. Fail if new criteria / filters are not applied correctly. Fail if there is insufficient error management (e.g. no error message when no flights match criteria, etc).  Pass otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F. 101 |
| Test Case ID | TCF. 101.03 |
| Test Case Name | Manual Test: View a specific flight manifest. |
| Test Steps and Test Data | Select "View Flight Manifests" from menu. Fill out criteria fields and submit form. Select a flight from list of flight manifests. |
| Expected Result | Correct flight manifest is displayed for the selected flight. |
| Pass / Fail Criteria | Pass if the correct flight manifest is displayed. Fail if no proper error message displayed when manifest is unavailable/inaccessible. Fail if manager cannot view flight manifest they should have access to. Fail if there is insufficient error management |

|  |  |
| --- | --- |
| Requirement ID | F. 101 |
| Test Case ID | TCF. 101.04 |
| Test Case Name | Manual Test: Return to manifest list and select another flight manifest |
| Test Steps and Test Data | Select "View Flight Manifests" from menu. Fill out criteria fields and submit form. Select a flight from list of flight manifests. Click "Return to manifest list". Select another flight from the list. |
| Expected Result | Correct flight manifest is displayed for the selected flight. |
| Pass / Fail Criteria | Pass if correct flight manifest is displayed. Fail if manager cannot view flight manifest they should have access to. Fail if there is insufficient error management (e.g. when manifest is unavailable/inaccessible). |

|  |  |
| --- | --- |
| Requirement ID | F. 101 |
| Test Case ID | TCF. 101.05 |
| Test Case Name | Manual Test: Viewing details of a seat on flight manifest. |
| Test Steps and Test Data | Select "View Flight Manifests" from menu. Fill out criteria fields and submit form. Select a flight from list of flight manifests. Select a specific seat on the manifest. |
| Expected Result | Correct passenger's name is displayed, along with their meal plan, and seating class. |
| Pass / Fail Criteria | Fail if incorrect seat code displayed. Fail if incorrect passenger name displayed. Fail if incorrect meal plan for the passenger. Fail if incorrect seating class for the passenger. Fail if there is insufficient error management.  Pass otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F. 101 |
| Test Case ID | TCF. 101.06 |
| Test Case Name | Manual Test: Return to manifest and view another seat’s details. |
| Test Steps and Test Data | Select "View Flight Manifests" from menu. Fill out criteria fields and submit form. Select a flight from list of flight manifests. Select a specific seat on the manifest. Click "Return to Manifest" after selecting a seat. Select another seat from manifest. |
| Expected Result | Correct passenger's name is displayed, along with their meal plan, and seating class. |
| Pass / Fail Criteria | Fail if incorrect manifest is returned. Fail if incorrect seat code displayed. Fail if incorrect passenger name displayed. Fail if incorrect meal plan for the passenger. Fail if incorrect seating class for the passenger. Fail if there is insufficient error management.  Pass otherwise. |

### View Flight Statistics

|  |  |
| --- | --- |
| Requirement ID | F.102 |
| Test Case ID | F.102.0 |
| Test Case Name | Manual Test: View flight statistics search page. |
| Test Steps and Test Data | Select view flight statistics from menu. |
| Expected Result | You are taken to the flight statistics search page. |
| Pass Fail Criteria | Pass if link leads to correct page.  Pass if search fields function correctly.  Fail otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F.102 |
| Test Case ID | F.102.1 |
| Test Case Name | Manual Test: View flight statistics page. |
| Test Steps and Test Data | Go to the flight statistics search page, and specify search criteria in Search / Filter flights page. |
| Expected Result | User is taken from the flight statistics search page to the flight statistics page and the correct statistics (historical and most popular destination) are displayed. |
| Pass Fail Criteria | Fail if flight statistic is not reached.  Fail if statistics are displayed incorrectly.  Fail if statistics did not follow search criteria.  Pass otherwise. |

|  |  |
| --- | --- |
| Requirement ID | F.102 |
| Test Case ID | F.102.2 |
| Test Case Name | Manual Test: View historical statistics |
| Test Steps and Test Data | Go to flight statistics search page.  Enter criteria into search fields and submit form. |
| Expected Result | You are taken to the flight statistics page, and historical statistics are displayed. |
| Pass Fail Criteria | Fail if past statistics are displayed incorrectly.  Fail if past statistics are missing.  Fail if link does not lead to correct page.  Pass otherwise. |

### Modify Reservation on Behalf of Customer

|  |  |
| --- | --- |
| Requirement ID | F.103 |
| Test Case ID | TC F.103.0 |
| Test Case Name | Manual Test: Manager view bookings page loading. |
| Test Steps and Test Data | Click view booking option from top menu.  Manager enters customer’s email address. |
| Expected Result | Page for manager to view customers’ bookings are displayed. |
| Pass Fail Criteria | Pass if page for manager to view customers’ bookings is displayed. Fail if other pages are loaded after click. |

|  |  |
| --- | --- |
| Requirement ID | F.103 |
| Test Case ID | TC F.103.1 |
| Test Case Name | Manual Test: Manager updates bookings. |
| Test Steps and Test Data | Click update tickets menu from the side.  Enter preferred options.  Click update tickets. |
| Expected Result | Page for manager to update customers’ booking is displayed. |
| Pass Fail Criteria | Pass if page for manager to update customers’ booking is displayed. Fail if other pages are loaded after click. |

### Cancel Reservation on Behalf of Customer

|  |  |
| --- | --- |
| Requirement ID | F.104 |
| Test Case ID | TC F.104.0 |
| Test Case Name | Manual Test: Manager view bookings page loading. |
| Test Steps and Test Data | Click view booking option from top menu.  Manager enters customer’s email address. |
| Expected Result | Page for manager to view customers’ bookings are displayed. |
| Pass Fail Criteria | Pass if page for manager to view customers’ bookings is displayed. Fail if other pages are loaded after click. |

|  |  |
| --- | --- |
| Requirement ID | F.104 |
| Test Case ID | TC F.104.1 |
| Test Case Name | Manual Test: Manager cancels bookings. |
| Test Steps and Test Data | Click cancel booking menu from top.  Click confirm cancellation button. |
| Expected Result | Cancellation is successful message is displayed. |
| Pass Fail Criteria | Pass if page for manager to cancel customers’ booking is displayed. Fail if other pages are loaded after click. |

## Non Functional Requirements

|  |  |
| --- | --- |
| Requirement ID | SEC.0 |
| Test Case ID | TC SEC.0.00 |
| Test Case Name | User data security |
| Test Steps and Test Data | Request Pod Ten for data about a particular customer. |
| Expected Result | Access to user data should not be granted without valid authorization such as a police warrant. |
| Pass Fail Criteria | Fail if access is granted without valid authorization. Fail if access is not granted with proper authorization. |

|  |  |
| --- | --- |
| Requirement ID | SEC.0 |
| Test Case ID | TC SEC.0.01 |
| Test Case Name | Policy regarding selling user data |
| Test Steps and Test Data | Request Pod Ten to setup a data sharing agreement with an advertising company. |
| Expected Result | Request must be denied as this is against Pod Ten company policy and user agreement. |
| Pass Fail Criteria | Fail if request is not denied. |

|  |  |
| --- | --- |
| Requirement ID | OPE.0 |
| Test Case ID | TC OPE.0.00 |
| Test Case Name | Transaction logging |
| Test Steps and Test Data | Open logs. Check if significant user transactions are logged. This includes user registration, reservation of flight tickets, printing of boarding pass, cancellation of reservation. |
| Expected Result | All significant transactions should be logged in a verbose manager, along with timestamp, user Id and prior and new values (if applicable) of database objects. |
| Pass Fail Criteria | Fail if any such transaction is not logged. Fail if logs are not understandable to someone with basic knowledge of system. Fail if logs do not include timestamp, user Id, and prior and new values (if applicable) |

|  |  |
| --- | --- |
| Requirement ID | OPE.1 |
| Test Case ID | TC OPE.1.00 |
| Test Case Name | Uptime |
| Test Steps and Test Data | Check for availability at random times between 6:00am and 1:00am. |
| Expected Result | The site should be up at all times, unless prior notification has been sent to users. |
| Pass Fail Criteria | Fail if the site is not up at any given time between 6:00am and 1:00am without prior notification. |

|  |  |
| --- | --- |
| Requirement ID | INT.0 |
| Test Case ID | TC INT.0.00 |
| Test Case Name | Platform independence |
| Test Steps and Test Data | Try using various features of the website on Google Chrome on different operating systems including Windows, Mac OSX, Ubuntu. |
| Expected Result | The site should run properly on each operating system. |
| Pass Fail Criteria | Fail if the site is not loaded properly on any operating system, or if the site requests the user to switch to another operating system. |

|  |  |
| --- | --- |
| Requirement ID | PER.0 |
| Test Case ID | TC PER.0.00 |
| Test Case Name | Responsiveness |
| Test Steps and Test Data | Try using various features of the website. Measure the response time of each action. |
| Expected Result | No user action should have a response time of more than 0.1 seconds. |
| Pass Fail Criteria | Fail if any user action results in response time of more than 0.1 seconds. |

## User Login Cases

|  |  |
| --- | --- |
| Test Case ID | TC.00 |
| Test Case Name | Manual Test: Valid ID and Valid Password. |
| Test Steps | Go to homepage. Login with valid user name and password. |
| Expected Result | User is logged into the system and is directed to homepage. |
| Pass Fail Criteria | Pass if user is logged in and is redirected to search page. User’s session must also be created. Fail otherwise. |

|  |  |
| --- | --- |
| Test Case ID | TC.01 |
| Test Case Name | Manual Test: User Name Or Password Not Filled In |
| Test Steps | Go to homepage. Input only user name, or only password, or neither and try to login. |
| Expected Result | Return an error message stating user is missing user name, password or both. |
| Pass Fail Criteria | Pass if the error message is displayed and the user is not logged in. Fail if the user is logged in without password, or error message not displayed. |

|  |  |
| --- | --- |
| Test Case ID | TCF.005.02 |
| Test Case Name | Manual Test: Valid User Name, Invalid Password |
| Test Steps | Go to homepage. Input user name and incorrect password and try to login. |
| Expected Result | Display an error message stating that credentials are incorrect, user is not logged in. |
| Pass Fail Criteria | Pass if user is not logged in and error message is displayed. Fail if user is logged in and error message not displayed. |

|  |  |
| --- | --- |
| Test Case ID | TCF.005.03 |
| Test Case Name | Manual Test: User Does Not Exist |
| Test Steps | Go to homepage. Input invalid email address and try to login. |
| Expected Result | Display an error message stating that credentials are incorrect. |
| Pass Fail Criteria | Pass if the error message is displayed.  Fail if the error message is not displayed or the user is logged in under invalid user id. |

|  |  |
| --- | --- |
| Test Case ID | TCF.005.04 |
| Test Case Name | Manual Test: Create new user |
| Test Steps | Go to homepage.  Click on ‘New User?’.  Input new user name and password and re-enter the password. |
| Expected Result | User account is created with input user name and password. |
| Pass Fail Criteria | Pass if account is created and user can log in. User must also be logged in and redirected to search page. Session must be created.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case ID | TCF.005.05 |
| Test Case Name | Manual Test: Create New User – Missing Input Field |
| Test Steps | Go to homepage. Select ‘Create New User’.  Input either a user name, or a password, or neither. |
| Expected Result | Return an error message stating user is missing user name, password or both. |
| Pass Fail Criteria | Pass if the error message is displayed and the new user account is not created. Fail if the new user account is created with user name alone, or error message is not displayed. |

|  |  |
| --- | --- |
| Test Case ID | TCF.005.06 |
| Test Case Name | Manual Test: Create New User – User Name Already Exists |
| Test Steps | Go to homepage. Select ‘Create New User’.  Use an email address that already exists in system. |
| Expected Result | Display an error message stating that the email address has already been used. |
| Pass Fail Criteria | Pass if user account is not created under duplicate email address, and error message is displayed. Fail if user account is created under duplicate email address, and error message not displayed. |

|  |  |
| --- | --- |
| Test Case ID | TCF.005.07 |
| Test Case Name | Manual Test: Test user action while not logged in |
| Test Steps | Enter a valid URL with valid parameters into address bar. Ensure that user is not logged in. |
| Expected Result | User is redirected to login page. |
| Pass Fail Criteria | Pass if the user is redirected and the user is not allowed to view result of URL that was typed in. |

## Unit Tests

### User Service Tests

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Create new test user |
| Test Steps and Test Data | Pre-condition: emailAddress does not exist in database yet  Test Data:  String name = "tester";  String password = "test1234";  String emailAddress = "test@pod10.com";  Invoke createUser method of User Service using test data |
| Expected Result | New user object should be created in the database |
| Pass / Fail Criteria | Pass if new user successfully created.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Attempt to reuse email address when create new user |
| Test Steps and Test Data | Pre-condition: emailAddress already exists in database  Invoke createUser method of User Service when emailAddress data was used before |
| Expected Result | New user should NOT be successfully created |
| Pass / Fail Criteria | Pass if new user not successfully created.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Login user with correct password |
| Test Steps and Test Data | Pre-condition: Associated user object has been created  Invoke loginCheck method of User Service inputting user and correct password |
| Expected Result | Login should be successful |
| Pass / Fail Criteria | Pass if login check was successful.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Login user with incorrect password |
| Test Steps and Test Data | Pre-condition: Associated user object has been created  Invoke loginCheck method of User Service inputting user and INCORRECT password |
| Expected Result | Login should NOT be successful |
| Pass / Fail Criteria | Pass if login check was not successful.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Get User object by correct ID |
| Test Steps and Test Data | Pre-condition: User object has been created and associated userID, name, email address is known  Invoke getUserById method of User Service inputting correct user id |
| Expected Result | User object successfully retrieved and member variable (name, email address) is correct |
| Pass / Fail Criteria | Pass if user object successfully retrieved and has correct values.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Get User ID by email address |
| Test Steps and Test Data | Pre-condition: User object has been created and associated userID, name, email address is known  Invoke getUserIdByEmail method of User Service inputting test emailAddress |
| Expected Result | UserID associated with emailAddress should be returned correctly |
| Pass / Fail Criteria | Pass if correct UserID is returned.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Get User points |
| Test Steps and Test Data | Pre-condition: User object has been created  Invoke getPoints method of User Service inputting test userID |
| Expected Result | Points returned should match entry in table |
| Pass / Fail Criteria | Pass if points returned matches table entry.  Fail otherwise. |

### Reservation Service Unit Tests

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Add booking for user |
| Test Steps and Test Data | Invoke addBooking method of Reservation Service inputting valid user ID |
| Expected Result | Booking successfully added and valid booking id returned |
| Pass / Fail Criteria | Pass if booking successfully added with valid bookingID.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Get current booking function |
| Test Steps and Test Data | Pre-condition: Known bookingID with associated userID  Invoke getCurrentBooking method of Reservation Service inputting valid bookingID |
| Expected Result | Non-null Booking object returned with correct userID |
| Pass / Fail Criteria | Pass if non-null Booking object returned with correct userID.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Get Booking Lists by UserID |
| Test Steps and Test Data | Pre-condition: Known bookingID with associated userID  Invoke getBookingList method of Reservation Service inputting known userID  Iterated through returned list to find known bookingID |
| Expected Result | Known bookingID/userID pair is returned in the Booking list |
| Pass / Fail Criteria | Pass if bookingID/userID pair is returned in Booking List.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Add tickets for user booking |
| Test Steps and Test Data | Pre-condition: A valid user and booking has been created  Invoke addTickets method of Reservation Service inputting user info and booking id |
| Expected Result | Add tickets executed successfully |
| Pass / Fail Criteria | Pass if add tickets executed successfully.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Get tickets list for user booking |
| Test Steps and Test Data | Pre-condition: Tickets (with known attributes) have been added to known bookingID  Invoke getTicketSetList method of Reservation Service inputting bookingID  Iterate through returned ticket list and match the tickets’ attributes with known info |
| Expected Result | Ticket list contains correct number of tickets with correct information |
| Pass / Fail Criteria | Pass if ticket list contains correct number of tickets with correct information.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Add billing for user booking |
| Test Steps and Test Data | Pre-condition: A valid user and booking has been created  Test Data:  String creditCardNum = "4500123456785555";  **int** amount = 2500;  Invoke addBilling method of Reservation Service inputting user info and test data |
| Expected Result | Add billing operation should be executed successfully |
| Pass / Fail Criteria | Pass if add billing operation was successful.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Get billing for user booking |
| Test Steps and Test Data | Pre-condition: A valid user, booking, and billing has been created  Invoke getBilling method of Reservation Service |
| Expected Result | Billing object should be returned with correct values (amount, name, credit card number) |
| Pass / Fail Criteria | Pass if billing object returned correctly.  Fail otherwise. |

### Flight Service Unit Tests

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Get flights list |
| Test Steps and Test Data | Pre-condition: Flights table is populated with any flight data  Invoke getFlightSelection method of Flight Service |
| Expected Result | List of flights object should be returned and not null |
| Pass / Fail Criteria | Pass if list of flights object returned and not null.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Search flight function |
| Test Steps and Test Data | Test data:  *listOfFlights* - Obtained using getFlightSelection  Extract a flight from *listOfFlights* and get origin, destination, and departure time variables  Invoke getFlights method of Flight Service inputting the variables obtained in previous step  Repeat process for every flight |
| Expected Result | Search flight function should return correct flightID based on parameters inputted for every flight |
| Pass / Fail Criteria | Pass if correct flightID returned for every flight searched.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Get Flight Statistics |
| Test Steps and Test Data | Invoke getStatistics method of Flight Service |
| Expected Result | Valid (non-null) Statistics object is returned |
| Pass / Fail Criteria | Pass if non-null Statistics object is returned.  Fail otherwise. |

|  |  |
| --- | --- |
| Test Case Name | Unit Test: Get Flight Manifest |
| Test Steps and Test Data | Pre-condition: Known ticket for a given FlightID has been created before  Invoke getFlights method of Flight Service inputting FlightID  Verify known ticket is listed in returned ticket list (manifest) |
| Expected Result | Known ticket is listed in the list AND all tickets in the list have same flightID |
| Pass / Fail Criteria | Pass if known ticket is listed and all tickets have identical flightID.  Fail otherwise. |

# Appendix B: Box

In this Appendix are included two images of the box that the software will be sold in.

# 

# 

# Appendix C: Breakdown of workload

**Hours**

|  |  |  |  |
| --- | --- | --- | --- |
| **ASSIGNMENT #** | **TASK DESCRIPTION** | **TASK CATEGORY** | **DURATION (HOURS)** |
| **week 3** |  |  |  |
| Assignment 2 (Box) | Non functional req | Proposal documentation | 1 |
| Assignment 2 | Editing proposal | Proposal documentation | 3 |
| **week 4** |  |  |  |
| Assignment 2 | Editing text in intro/purpose etc | Proposal documentation | 3 |
| Assignment 3 | Deployment Architecture | Architecture design + documentation | 0.5 |
| **week 5** |  |  |  |
| Assignment 3 | Class Diagram | Architecture design + documentation | 2 |
| Assignment 3 | Review Sequence Diagrams | Architecture design + documentation | 2.5 |
| Assignment 3 | Editing of document | Documentation | 3 |
| Assignment 4 | Set up dev env | System administration and setup | 8 |
| **week 6** |  |  |  |
| Assignment 3/4 | Meet with TA regarding assignment 3 and 4 | Meet with TA | 0.5 |
| Assignment 4 | Non functional req test cases | Documentation | 1 |
| Assignment 4 | Editing test plan document | Documentation | 3 |
| **week 7** |  |  |  |
| Assignment 4 | Final editing test plan document | Documentation | 7 |
| Final Demo | Implementation of flight manifest use case | Implementation | 2.5 |
| Final Demo | Implementation of boarding pass use case | Implementation | 4 |
| **weel 8** |  |  |  |
| Final Demo | Fixing project structure | Implementation | 10 |
| **week 9** |  |  |  |
| Final Demo | Implementation of user login | Implementation | 10 |
| **week 10** |  |  |  |
| Final Demo | Improving user login | Implementation | 4 |
| **week 11** |  |  |  |
| Final Demo & report | Scoping out tasks for following week | Project scoping | 2 |
| Final Demo | Boarding pass implementation | Implementation | 2 |
| **week 12** |  |  |  |
| Final Demo | Meeting with Shirley, Nathaniel | Design | 0.5 |
| Final report | Update requirements, architecture, testing for print boarding pass and non-funct req | Documentation | 3 |
| Final report | Update requirements, architecture, testing for user login and print boarding pass | Documentation | 3 |
| Final report | Updated interesting design/architecture aspects | Documentation | 4 |
| Final report | Compilation/editing | Documentation | 4 |
| Final report | Compilation/editing | Documentation | 4 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  | **Total: 87.5** |

**A.2 Ryan Randhawa**

|  |  |  |  |
| --- | --- | --- | --- |
| **ASSIGNMENT #** | **TASK DESCRIPTION** | **TASK CATEGORY** | **DURATION (HOURS)** |
| **week 5** |  |  |  |
| Assignment 2 | Formatting and editing proposal | Documentation | 2.5 |
| Assignment 3 | Sequence Diagram | Architecture | 1 |
| Assignment 3 | Compiling Document | Documentation | 0.75 |
| Assignment 4 | Search Flight Test Case | Documentation | 1 |
| **week 7** |  |  |  |
|  | Setting up repository | Repository | 1 |
| Final Demo | Implementation of flight manifest use case | Implementation | 2.5 |
| **week 8** |  |  |  |
|  | Adding Jar files and updating code | Implementation | 0.5 |
|  |  |  | 3 |
|  | View Boarding Pass Use Case |  | 4 |
|  | View Boarding Pass Use Case (with class diagram changes) | Implementation | 4 |
| **week 10** |  |  |  |
|  | View Booking use case |  | 4 |
|  | Changes based on new class diagram |  | 5 |
|  | View booking use case |  |  |
| **week 11** |  |  |  |
|  | Cancel bookings use case |  | 2 |
|  | Cancel Bookings use case |  | 3 |
|  | Adding login code |  | 0.5 |
|  | View Bookings Use Case - fixing FTL |  | 1 |
|  | Updating Modify Bookings and Cancel Bookings and Manage Booking Documentation | Documentation | 2.5 |
| **week 12** |  |  |  |
|  | View Bookings Use Case showing tickets |  | 6 |
|  | View Bookings Sequence Diagram |  | 1 |
|  | Fixing Cancel Booking and View Booking Use Cases |  | 3 |
|  | Added Staff versions of Cancel Bookings, Modify Bookings, and View Bookings |  | 1.5 |
|  | Fixing an error with Billings with Points |  | 1 |
|  | Bugfix with Cancel Booking and Points |  |  |
|  | Staff View Booking List Action | Implementation | 2 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  | **Total: 52.75** |

**A.3** **Kevin Chow**

|  |  |  |  |
| --- | --- | --- | --- |
| **ASSIGNMENT #** | **TASK DESCRIPTION** | **TASK CATEGORY** | **DURATION (HOURS)** |
| **week 4** |  |  |  |
| Assignment #3 | View Flight Manifest (manager) use case diagram | Documentation | 1.5 |
| Assignment#3 | Purpose + Scope + Overview | Documentation | 0.6 |
| **week 5** |  |  |  |
| Assignment#3 | Purpose + Scope + Overview | Documentation | 0.6 |
| Assignment#3 | Sequence diagram modification | Documentation | 0.3 |
| **week 6** |  |  |  |
| Assignment #3 | Test cases | Documentation | 1 |
| **week 8** |  |  |  |
| Assignment #4 | Setting up dev environment | Set up | 5 |
| Assignment #4 | Writing test cases | Testing | 1.5 |
| Assignment #4 | Login screen | Development | 2.5 |
| **week 9** |  |  |  |
| Assignment #4 | Login screen | Development | 2 |
| **week 11** |  |  |  |
| Assignment #4 | Select Manifest | Development | 3 |
| Assignment #4 | Select Manifest | Development | 2 |
| Assignment #4 | Select Manifest | Development | 2 |
| **week 12** |  |  |  |
| Assignment #4 | Select Manifest | Development | 1 |
| Assignment #4 | Select Manifest | Development | 1 |
| Assignment #4 | Select Manifest | Development | 2 |
| Assignment #4 | Purpose + Scope + Overview | Final Report | 1.5 |
| Assignment #4 | Purpose + Scope + Overview | Final Report | 1.5 |
| Assignment #4 | Update sequence diagram | Final Report | 1 |
| Assignment #4 | Update past docs | Final Report | 3 |
| Assignment #4 | Bugfixing | Final Demo | 1 |
| Assignment #4 | Glossary | Final Report | 1 |
| Assignment #4 | Glossary | Final Report | 1.5 |
| Assignment #4 | Proofreading final report | Final Report | 2.5 |
| Assignment #4 | Proofreading final report | Final Report | 2.5 |
|  |  |  |  |
|  |  |  |  |
|  |  |  | **Total: 41.5** |

**A.4 Nathaniel Sham**

|  |  |  |  |
| --- | --- | --- | --- |
| **ASSIGNMENT #** |  | **TASK CATEGORY** | **DURATION (HOURS)** |
| **wekk 3** |  |  |  |
| Assignment 2 | sketch some UI | Documentation | 1 |
| Assignment 2 | sketch some UI | Documentation | 2 |
| **week 4** |  |  |  |
| Assignment 2 | update UI sketch (more specific) | Documentation | 1.5 |
| Assignment 2 | sw requirements doc header | Documentation | 0.25 |
| **week 5** |  |  |  |
| A3 | class diagram | Documentation | 2 |
| A4 | test cases - english | Documentation | 1.5 |
| **week 6** |  |  |  |
| A4 | test cases - refactor | Documentation | 2.5 |
| A4 | test cases - refactor | Documentation | 0.5 |
| **week 7** |  |  |  |
| A4 | edit test document | Documentation | 2 |
| A4 | consistency, fix some test cases | Documentation | 1 |
| A4 | consistency, fix some test cases | Documentation | 2.6 |
| **week 8** |  |  |  |
|  | html | UI | 1 |
|  | css | UI | 2 |
|  | html | UI | 0.5 |
| **week 9** |  |  |  |
|  | css | UI | 2 |
|  | html, css | UI | 2 |
|  | html, css, jquery | UI | 2 |
| **week 10** |  |  |  |
|  | html, css, js/jquery | UI | 6.5 |
|  | error pages | UI | 2 |
| **week 11** |  |  |  |
|  | javascript - empty inputs | UI | 1.5 |
|  | html, css, javascript | UI | 5.5 |
|  | html, javascript | UI | 2 |
|  | html, javascript | UI | 2 |
|  | html, css | UI | 1 |
|  | html | UI | 0.5 |
| **week 12** |  |  |  |
|  | html, css | UI | 1 |
|  | html | UI | 4 |
|  | viewbookings | UI | 2 |
|  | html | UI | 1 |
|  | html, css (tabs) | UI | 1 |
|  | html | UI | 2 |
|  | html, js (some staff stuff) | UI | 1.5 |
|  | html, css, js/jquery, java | mainly staff UI | 4.5 |
|  | html, css, js/jquery, java | mainly staff UI | 4 |
|  | html, css, js/jquery, java | mainly staff UI | 2.5 |
|  | html, css, js/jquery | UI | 4 |
|  |  |  | **Total: 74.85** |

**A5. Yu Han**

|  |  |  |  |
| --- | --- | --- | --- |
| **ASSIGNMENT #** | **TASK DESCRIPTION** | **TASK CATEGORY** | **DURATION (HOURS)** |
| **week 3** |  |  |  |
| Assignment 2 | Compile proposal | Documentation | 3 |
| Assignment 2 | Compile proposal | Documentation | 0.5 |
| **week 4** |  |  |  |
| Assignment 3 | sequence diagram(modify reservation) | Architecture Design | 1.5 |
| **week 5** |  |  |  |
| Assignment 3 | sd review and modify | Architecture Design | 1 |
| **week 6** |  |  |  |
| Assignment 4 | test cases description | Documentation | 1 |
| Assignment 4 | test cases description | Documentation | 2.5 |
|  | class implementation | dev | 3.2 |
| **week 7** |  |  |  |
| Use Case | Use case:searchFlights | dev | 9.5 |
| **week 8** |  |  |  |
| use Case | Use case: reserveFlights | dev | 2 |
| use Case | Use case: reserveFlights | dev | 8.5 |
| **week 10** |  |  |  |
| use case | User case:reserverFlights | dev | 2 |
| use case | java doc | Documentation | 1 |
| use case | java doc | Documentation | 1 |
| **week 11** |  |  |  |
| use case | Use case | dev | 4 |
| use case | return flights | dev | 4 |
| use case | return flights | dev | 3.3 |
| use case | use case doc | Documentation | 1 |
| **week 12** |  |  |  |
| debug | debug boardingpass | dev | 6.5 |
| debug | debug transactions | dev | 5 |
| debug | debug transactions | dev | 1.5 |
| print log | dev | dev | 0.5 |
| print log | dev | dev | 0.5 |
| final document | documentation | Documentation | 3.5 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  | **Total: 66.5** |

**A.6 Shirley**

|  |  |  |  |
| --- | --- | --- | --- |
| **ASSIGNMENT #** | **TASK DESCRIPTION** | **TASK CATEGORY** | **DURATION**  **(HOURS)** |
| **week 3** |  |  |  |
| Assignment 2 | Use case details for customer | Documentation | 2.5 |
| Assignment 2 | Use case details for customer - edit | Documentation | 0.5 |
| **week 4** |  |  |  |
| Assignment 3 | Sequence Diagram for View and Print Boarding Passes | Documentation | 1.5 |
| **week 5** |  |  |  |
| Assignment 3 | Sequence Diagram modification | Documentation | 0.75 |
| Assignment 3 | Update class diagram | Documentation | 2.5 |
| **week 6** |  |  |  |
| Assignment 4 | Test Case Description | Documentation | 2.5 |
| Assignment 4 | Environment setup | Documentation | 0.25 |
| Assignment 4 | Re-write requirements | Documentation | 0.5 |
|  | Test Case Description | Documentation | 3.5 |
|  | Environment setup | Development | 2 |
| **week 7** |  |  |  |
|  | Use case: search flights | Development | 9.5 |
| **week 8** |  |  |  |
|  | Use case: search flights | Development | 4 |
|  | Use case: reserve flight | Development | 2 |
| **week 9** |  |  |  |
|  | Use case: search flights | Development | 8.5 |
|  | Use case: search and reserve | Development | 6 |
| **week 11** |  |  |  |
|  | User authentication in search and reserve | Development | 4 |
|  | Search | Development | 2 |
|  | Sequence diagram | Documentation | 1 |
|  | Class/sequence diagram | Documentation | 5.5 |
| **week 12** |  |  |  |
|  | Database change | Development | 2.5 |
|  | Staff sign in / bug fix | Development | 1.5 |
|  | Pay by points bug fix | Development | 1.5 |
|  | Cancel booking bug fix | Development | 1.5 |
|  | Class/sequence diagram | Documentation | 2.5 |
|  | Testing / bug fix | Development | 2 |
|  | Code cleanup / logs | Development | 2 |
|  | Testing / bug fix with Nathaniel | Development | 3 |
|  | Design change(sessionToken) | Development | 3.5 |
|  |  |  | **Total : 79** |

**A.7 Yonatan**

|  |  |  |  |
| --- | --- | --- | --- |
| **ASSIGNMENT #** | **TASK DESCRIPTION** | **TASK CATEGORY** | **DURATION (HOURS)** |
| **week 3** |  |  |  |
| Assignment #2 | Use case - Airplane staff/manager | Documentation | 3 |
| **week 4** |  |  |  |
| Assignment #3 | Sequence diagram - Airplane staff/managers | Documentation | 3 |
| **week 5** |  |  |  |
| Assignment #3 | Sequence diagram updates | Documentation | 2 |
|  | JUnit trial |  | 2 |
| **week 6** |  |  |  |
| Assignment #4 | Test case doc | Documentation | 1.5 |
| Assignment #4 | Test case doc | Documentation | 3 |
| **week 7** |  |  |  |
| Assignment #4 | Test case doc | Documentation | 1.5 |
| Final Demo | Unit test skeleton | Testing/ Development | 1 |
| **week 9** |  |  |  |
| Final Demo | User unit test | Testing/ Development | 3 |
| Final Demo | Manual test | Testing | 2 |
| **week 10** |  |  |  |
| Final Demo | Reservation unit test | Testing/ Development | 2.5 |
| Final Demo | Flight Unit test | Testing/ Development | 2 |
| Final Demo | Manual test | Testing | 0.5 |
| Final Demo | Unit test add | Testing/ Development | 3 |
| **week 11** |  |  |  |
| Final Demo | Unit test add | Testing/ Development | 2.5 |
| Final Demo | Manual test | Testing | 1.5 |
| **week 12** |  |  |  |
| Final Demo | Unit test add | Testing/ Development | 2 |
| Final Demo | Manual test | Testing | 2 |
| Final Demo | Final doc - test plan | Documentation | 1.5 |
| Final Demo | Unit test cleanup | Testing | 1 |
| Final Demo | Final doc - test plan | Documentation | 2 |
| Final Demo | Unit test cleanup | Testing | 1.5 |
| Final Demo | Manual test | Testing | 1 |
| Final Demo | Final doc - test plan | Documentation | 2 |
|  |  |  | **Total :47** |

**A.8 Scott**

|  |  |  |  |
| --- | --- | --- | --- |
| **ASSIGNMENT** | **TASK DESCRIPTION** | **TASK CATEGORY** | **DURATION (HOURS)** |
| **week 3** |  |  |  |
| Assignment 2 | Box design | Documentation | 2 |
| Assignment 2 | Writing document | Documentation | 2 |
| **week 4** |  |  |  |
| Assignment 3 | Sequence Diagrams + UML Diagram | Documentation | 2 |
| Assignment 3 | document editing | Documentation | 1 |
| **week 5-12** |  |  |  |
| Assignment 4 | Purpose Scope, Overview | Documentation | 2.5 |
| Assignment 4 | Test Case - Reserve Flight | Documentation | 2 |
| Assignment 4 | Test Case - Reserve Flight | Documentation | 3 |
| Final Demo | Environment Setup | Development | 2 |
| Final Demo | Implementation of flight manifest use case | Development | 2.5 |
| Final Demo | Implementation of user login - initial | Development | 3 |
| Final Demo | Implementation of user login - initial | Development | 1 |
| Final Demo | Implementation of user login - initial | Development | 2 |
| Final Demo | Implementation select flight manifest - initial | Development | 4 |
| Final Demo | Implementation select flight manifest - initial | Development | 2 |
| Final Demo | Implementation of view transaction history | Development | 4.5 |
| Final Demo | Redo sequence diagram and uml | Development | 1.5 |
| Final Demo | Project Results | Documentation | 0.5 |
| Final Demo | User Manual - user | Documentation | 4.5 |
| **Total** |  |  | **42** |

**A.9 Loui**

|  |  |  |  |
| --- | --- | --- | --- |
| **ASSIGNMENT #** | **TASK DESCRIPTION** | **TASK CATEGORY** | **DURATION (HOURS)** |
| **week 4** |  |  |  |
| Assignment 3 | Sequence Diagram for manage payment | Documentation | 1 |
| **week 5** |  |  |  |
| Assignment 3 | Sequence Diagram modification | Documentation | 0.5 |
| Assignment 4 | Test Cases for manage payment | Documentation | 0.5 |
| **week 6** |  |  |  |
| Assignment 4 | Test Cases for manage payment | Documentation | 1.25 |
| Assignment 4 | Test Cases for manage payment | Documentation | 1 |
| Assignment 4 | Modify Test Cases for manage payment | Documentation | 0.5 |
| **week 9** |  |  |  |
| Assignment 4 | Creating .css and .html | Documentation | 1 |
| Assignment 4 | Creating .css and .html | Documentation | 2 |
| **week 11** |  |  |  |
| Assignment 4 | Modify Ticket and View Statistics use cases | Development | 6 |
| Assignment 4 | Modify Ticket and View Statistics use cases | Development | 4.5 |
| Assignment 4 | Modify Ticket and View Statistics use cases | Development | 3 |
| Assignment 4 | Modify Ticket and View Statistics use cases | Development | 5.5 |
| Assignment 4 | Modify Ticket and View Statistics use cases | Documentation | 2 |
| **week 12** |  |  |  |
| Assignment 4 | View Statistics ftl file fixes | Development | 1 |
| Assignment 4 | View Statistics Sequence Diagram | Documentation | 1 |
| Assignment 4 | Sequence Diagrams | Documentation | 4.5 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  | **Total: 35.25** |

**A.10 Team Meeting**

|  |  |  |
| --- | --- | --- |
|  | **Duration (hours)** | **Time** |
| **Meeting #1 (11/09/2012)** | 1.5 | **week2** |
| **Meeting #2 (18/09/2012)** | 1.7 | **week3** |
| **Meeting #3 (25/09/2012)** | 1.5 | **week4** |
| **Meeting #4 (02/10/2012)** | 0.5 | **week5** |
| **Meeting #5 (09/10/2012)** | 1.5 | **week6** |
| **Meeting #6 (12/10/2012)** | 4 | **week6** |
| **Meeting #7 (13/10/2012)** | 4 | **week6** |
| **Meeting #8 (16/10/2012)** | 1.5 | **week7** |
| **Meeting #9 (23/10/2012)** | 1.5 | **week8** |
| **Meeting #10 (26/10/2012)** | 1.5 | **week8** |
| **Meeting #11 (30/10/2012)** | 1.5 | **week9** |
| **Meeting #12 (06/11/2012)** | 1.8 | **week10** |
| **Meeting #13 (12/11/2012)** | 5.5 | **week 11** |
| **Meeting #14 (20/11/2012)** | 1 | **week12** |
|  |  |  |
|  |  |  |
|  | **Total:** | **29** |

# Appendix C: Code

BaseAction.java

package ubc.ece419.pod10.action;

import ubc.ece419.pod10.domain.User;

import ubc.ece419.pod10.service.UserService;

import com.opensymphony.xwork2.ActionSupport;

@SuppressWarnings("serial")

public class BaseAction extends ActionSupport {

public User user;

UserService userService = new UserService();

public BaseAction() {

super();

}

public String execute() {

return SUCCESS;

}

/\*\*

\* Authenticates a userId. Also sets user.

\* @param userId Identifies a user.

\* @return True is user is authenticated, false otherwise.

\*/

public Boolean authenticate(Long userId) {

user = userService.getUserById(userId);

return userService.hasValidSession(userId);

}

public User getUser() {

return user;

}

public void setUser(User user) {

this.user = user;

}

}

Billing.java

package ubc.ece419.pod10.domain;

import java.sql.Timestamp;

public class Billing {

public Billing() {

super();

// TODO Auto-generated constructor stub

}

public Billing(Long id, Long userId, Timestamp dateCreated,

Timestamp dateModified, Long bookingId, int amount, String name,

String creditCardNum, boolean isLoyaltyPointsUsed) {

super();

this.id = id;

this.userId = userId;

this.dateCreated = dateCreated;

this.dateModified = dateModified;

this.bookingId = bookingId;

this.amount = amount;

this.name = name;

this.creditCardNum = creditCardNum;

this.isLoyaltyPointsUsed = isLoyaltyPointsUsed;

}

private Long id;

private Long userId;

private Timestamp dateCreated;

private Timestamp dateModified;

private Long bookingId;

private int amount;

private String name;

private String creditCardNum;

private boolean isLoyaltyPointsUsed;

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public Long getUserId() {

return userId;

}

public void setUserId(Long userId) {

this.userId = userId;

}

public Timestamp getDateCreated() {

return dateCreated;

}

public void setDateCreated(Timestamp dateCreated) {

this.dateCreated = dateCreated;

}

public Timestamp getDateModified() {

return dateModified;

}

public void setDateModified(Timestamp dateModified) {

this.dateModified = dateModified;

}

public Long getBookingId() {

return bookingId;

}

public void setBookingId(Long bookingId) {

this.bookingId = bookingId;

}

public int getAmount() {

return amount;

}

public void setAmount(int amount) {

this.amount = amount;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getCreditCardNum() {

return creditCardNum;

}

public void setCreditCardNum(String creditCardNum) {

this.creditCardNum = creditCardNum;

}

public boolean isLoyaltyPointsUsed() {

return isLoyaltyPointsUsed;

}

public boolean getIsLoyaltyPointsUsed() {

return isLoyaltyPointsUsed;

}

public void setLoyaltyPointsUsed(boolean isLoyaltyPointsUsed) {

this.isLoyaltyPointsUsed = isLoyaltyPointsUsed;

}

}

BoardingPass.java

package ubc.ece419.pod10.domain;

import java.sql.Timestamp;

// The boarding pass object takes a Ticket, Booking, and Flight object to generate a boarding pass

public class BoardingPass {

public BoardingPass() {

super();

}

public BoardingPass(Ticket myTicket, Booking myBooking, Flight myFlight) {

super();

this.myTicket = myTicket;

this.myBooking = myBooking;

this.myFlight = myFlight;

}

private Ticket myTicket;

private Booking myBooking;

private Flight myFlight;

// Auto generated getters and setters

public Ticket getMyTicket() {

return myTicket;

}

public void setMyTicket(Ticket myTicket) {

this.myTicket = myTicket;

}

public Booking getMyBooking() {

return myBooking;

}

public void setMyBooking(Booking myBooking) {

this.myBooking = myBooking;

}

public Flight getMyFlight() {

return myFlight;

}

public void setMyFlight(Flight myFlight) {

this.myFlight = myFlight;

}

}

Booking.java

package ubc.ece419.pod10.domain;

import java.sql.Timestamp;

public class Booking {

public Booking() {

super();

}

public Booking(Long iD, Timestamp dateCreated, Timestamp dateModified,

Long userID, String passengerName, Long flightMappingID, Long cancelState) {

super();

this.id = iD;

this.dateCreated = dateCreated;

this.dateModified = dateModified;

this.userID = userID;

this.cancelState = cancelState;

}

private Long id;

private Timestamp dateCreated;

private Timestamp dateModified;

private Long userID;

private Long cancelState;

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public Timestamp getDateCreated() {

return dateCreated;

}

public void setDateCreated(Timestamp dateCreated) {

this.dateCreated = dateCreated;

}

public Timestamp getDateModified() {

return dateModified;

}

public void setDateModified(Timestamp dateModified) {

this.dateModified = dateModified;

}

public Long getUserID() {

return userID;

}

public void setUserID(Long userID) {

this.userID = userID;

}

public Long getCancelState() {

return cancelState;

}

public void setCancelState(Long cancelState) {

this.cancelState = cancelState;

}

}

BookingSet.java

package ubc.ece419.pod10.domain;

import java.sql.Timestamp;

import java.util.ArrayList;

import java.util.List;

public class BookingSet {

public BookingSet() {

super();

}

public BookingSet(Booking booking, List<Ticket> ticketList) {

super();

this.booking = booking;

this.ticketList = ticketList;

}

private Booking booking;

private List<Ticket> ticketList;

public Booking getBooking() {

return booking;

}

public void setBooking(Booking booking) {

this.booking = booking;

}

public List<Ticket> getTicketList() {

return ticketList;

}

public void setTicketList(List<Ticket> ticketList) {

this.ticketList = ticketList;

}

}

CancelBookingAction.java

package ubc.ece419.pod10.action.user;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Billing;

import ubc.ece419.pod10.domain.Ticket;

import ubc.ece419.pod10.service.ReservationService;

import com.opensymphony.xwork2.ActionSupport;

@SuppressWarnings("serial")

public class CancelBookingAction extends BaseAction{

String bookingId;

int cancelState;

String userId;

Billing bill = new Billing();

List<Ticket> ticketList = new ArrayList<Ticket>();

ReservationService reservationService = new ReservationService();

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (bookingId == null) {

return ERROR;

}

Long bookingIdLong = Long.parseLong(bookingId);

cancelState = reservationService.getBookingCancelState(bookingIdLong);

if (cancelState != 0){

bill = reservationService.getCurrentBilling(bookingIdLong);

ticketList = reservationService.getCurrentTickets(bookingIdLong);

}

return SUCCESS;

}

public String getBookingId() {

return bookingId;

}

public void setBookingId(String bookingId) {

this.bookingId = bookingId;

}

public Billing getBill() {

return bill;

}

public void setBill(Billing bill) {

this.bill = bill;

}

public List<Ticket> getTicketList() {

return ticketList;

}

public void setTicketList(List<Ticket> ticketList) {

this.ticketList = ticketList;

}

public ReservationService getReservationService() {

return reservationService;

}

public void setReservationService(ReservationService reservationService) {

this.reservationService = reservationService;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public int getCancelState() {

return cancelState;

}

public void setCancelState(int cancelState) {

this.cancelState = cancelState;

}

}

ConfirmCancelAction.java

package ubc.ece419.pod10.action.user;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Billing;

import ubc.ece419.pod10.domain.Ticket;

import ubc.ece419.pod10.service.ReservationService;

import com.opensymphony.xwork2.ActionSupport;

public class ConfirmCancelAction extends BaseAction {

String bookingId, billingId, userId;

int cancelBooking, cancelTickets, refundBill;

ReservationService reservationService = new ReservationService();

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (bookingId == null) {

return ERROR;

}

Long bookingIdLong = Long.parseLong(bookingId);

Long billingIdLong = Long.parseLong(billingId);

cancelBooking = reservationService.cancelBooking(bookingIdLong);

cancelTickets = reservationService.cancelTickets(bookingIdLong);

refundBill = reservationService.refundBill(billingIdLong);

return SUCCESS;

}

public String getBookingId() {

return bookingId;

}

public void setBookingId(String bookingId) {

this.bookingId = bookingId;

}

public String getBillingId() {

return billingId;

}

public void setBillingId(String billingId) {

this.billingId = billingId;

}

public int getCancelBooking() {

return cancelBooking;

}

public void setCancelBooking(int cancelBooking) {

this.cancelBooking = cancelBooking;

}

public int getCancelTickets() {

return cancelTickets;

}

public void setCancelTickets(int cancelTickets) {

this.cancelTickets = cancelTickets;

}

public int getRefundBill() {

return refundBill;

}

public void setRefundBill(int refundBill) {

this.refundBill = refundBill;

}

public ReservationService getReservationService() {

return reservationService;

}

public void setReservationService(ReservationService reservationService) {

this.reservationService = reservationService;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

}

CreateUserAction.java

package ubc.ece419.pod10.action.user;

import ubc.ece419.pod10.service.UserService;

import ubc.ece419.pod10.exceptions.UnableToCreateUserException;

import com.opensymphony.xwork2.ActionSupport;

public class CreateUserAction extends ActionSupport {

UserService userService = new UserService();

// passwordCopy and submit are unused variables. Workaround so that Tomcat/Struts2 stop throwing errors for unset variables.

String name, password, passwordCopy, emailAddress;

String errorMessage;

Long userId;

public String execute() {

if (name == null || password == null || emailAddress == null) {

errorMessage = "Please fill out all the fields";

return ERROR;

}

if (userService.isEmailAddressAlreadyUsed(emailAddress)) {

errorMessage = "This email address is already registered. Please use another one.";

return ERROR;

}

try {

userId = userService.createUser(name, password, emailAddress);

} catch (UnableToCreateUserException e) {

errorMessage = e.getMessage();

return ERROR;

}

return SUCCESS;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getEmailAddress() {

return emailAddress;

}

public void setEmailAddress(String emailAddress) {

this.emailAddress = emailAddress;

}

public String getErrorMessage() {

return errorMessage;

}

public void setErrorMessage(String errorMessage) {

this.errorMessage = errorMessage;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Long getUserId() {

return userId;

}

public void setUserId(Long userId) {

this.userId = userId;

}

public String getPasswordCopy() {

return passwordCopy;

}

public void setPasswordCopy(String passwordCopy) {

this.passwordCopy = passwordCopy;

}

}

Flight.java

package ubc.ece419.pod10.domain;

import java.sql.Timestamp;

public class Flight {

public Flight() {

super();

}

public Flight(Long id, Timestamp dateCreated, Timestamp dateModified,

String origin, String originName, String destination,

String destinationName, Timestamp departureTime,

Timestamp arrivalTime, Timestamp actualDepartureTime,

Timestamp actualArrivalTime, String airlineName,

Integer totalSeats, Integer availableSeats, Integer cost) {

super();

this.id = id;

this.dateCreated = dateCreated;

this.dateModified = dateModified;

this.origin = origin;

this.originName = originName;

this.destination = destination;

this.destinationName = destinationName;

this.departureTime = departureTime;

this.arrivalTime = arrivalTime;

this.actualDepartureTime = actualDepartureTime;

this.actualArrivalTime = actualArrivalTime;

this.airlineName = airlineName;

this.totalSeats = totalSeats;

this.availableSeats = availableSeats;

this.cost = cost;

}

private Long id;

private Timestamp dateCreated;

private Timestamp dateModified;

private String origin;

private String originName;

private String destination;

private String destinationName;

private Timestamp departureTime;

private Timestamp arrivalTime;

private Timestamp actualDepartureTime;

private Timestamp actualArrivalTime;

private String airlineName;

private Integer totalSeats;

private Integer availableSeats;

private Integer cost;

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public Timestamp getDateCreated() {

return dateCreated;

}

public void setDateCreated(Timestamp dateCreated) {

this.dateCreated = dateCreated;

}

public Timestamp getDateModified() {

return dateModified;

}

public void setDateModified(Timestamp dateModified) {

this.dateModified = dateModified;

}

public String getOrigin() {

return origin;

}

public void setOrigin(String origin) {

this.origin = origin;

}

public String getDestination() {

return destination;

}

public void setDestination(String destination) {

this.destination = destination;

}

public Timestamp getDepartureTime() {

return departureTime;

}

public void setDepartureTime(Timestamp departureTime) {

this.departureTime = departureTime;

}

public Timestamp getArrivalTime() {

return arrivalTime;

}

public void setArrivalTime(Timestamp arrivalTime) {

this.arrivalTime = arrivalTime;

}

public Timestamp getActualDepartureTime() {

return actualDepartureTime;

}

public void setActualDepartureTime(Timestamp actualDepartureTime) {

this.actualDepartureTime = actualDepartureTime;

}

public Timestamp getActualArrivalTime() {

return actualArrivalTime;

}

public void setActualArrivalTime(Timestamp actualArrivalTime) {

this.actualArrivalTime = actualArrivalTime;

}

public String getAirlineName() {

return airlineName;

}

public void setAirlineName(String airlineName) {

this.airlineName = airlineName;

}

public Integer getTotalSeats() {

return totalSeats;

}

public void setTotalSeats(Integer totalSeats) {

this.totalSeats = totalSeats;

}

public Integer getAvailableSeats() {

return availableSeats;

}

public void setAvailableSeats(Integer availableSeats) {

this.availableSeats = availableSeats;

}

public Integer getCost() {

return cost;

}

public void setCost(Integer cost) {

this.cost = cost;

}

public String getOriginName() {

return originName;

}

public void setOriginName(String originName) {

this.originName = originName;

}

public String getDestinationName() {

return destinationName;

}

public void setDestinationName(String destinationName) {

this.destinationName = destinationName;

}

}

FlightService.java

package ubc.ece419.pod10.service;

import com.opensymphony.xwork2.ActionSupport;

import java.sql.\*;

import java.util.\*;

import org.omg.CORBA.Request;

import ubc.ece419.pod10.domain.Flight;

import ubc.ece419.pod10.domain.Ticket;

import ubc.ece419.pod10.domain.User;

import ubc.ece419.pod10.domain.Statistics;

public class FlightService extends ActionSupport {

private static final long serialVersionUID = 1L;

private String message;

String destination, destination2;

String flightId, flightId2, flightManifest\_flightId;

// the following fields are used for search for flights use case

String origin, departureTime, availableSeats;

//List<String> listofFlightsStr;

//Flights currentFlight;

List<Flight> listOfFlights;

List<Ticket> listOfTickets;

User user;

private static final String URL = "jdbc:mysql://localhost:3306/pod10\_db?zeroDateTimeBehavior=convertToNull";

private static final String DRIVER\_NAME = "com.mysql.jdbc.Driver";

private static final String DB\_USER\_NAME = "root";

private static final String DB\_PASSWORD = "ECE419\_pod10";

Connection con = null;

Statement stmt = null;

public FlightService() {

super();

try {

Class.forName(DRIVER\_NAME).newInstance();

con = DriverManager.getConnection(URL, DB\_USER\_NAME, DB\_PASSWORD);

} catch(InstantiationException e) {

e.printStackTrace();

} catch(IllegalAccessException e) {

e.printStackTrace();

} catch(ClassNotFoundException e) {

e.printStackTrace();

} catch (SQLException e) {

e.printStackTrace();

}

}

public String getFlightManifest\_flightId() {

return flightManifest\_flightId;

}

public List<Ticket> getListOfTickets() {

return listOfTickets;

}

public void setListOfTickets(List<Ticket> listOfTickets) {

this.listOfTickets = listOfTickets;

}

public void setFlightManifest\_flightId(String flightManifest\_flightId) {

this.flightManifest\_flightId = flightManifest\_flightId;

}

/\*

Old code, delete this later

public String getFlightManifest(String ) {

// Query database for a list of flights matching the flight code

try {

Class.forName(driverName).newInstance();

con = DriverManager.getConnection(url, userName, password);

stmt = con.createStatement();

ResultSet rs = stmt.executeQuery("SELECT \* FROM tickets WHERE flight\_id = flightManifest\_flightId");

ps = con.prepareStatement("SELECT \* FROM Flights WHERE origin = ? AND destination = ? AND departure\_time > ? AND available\_seats >= ?");

ps.setString(1, origin);

flightManifest\_flightId

if (rs == null) {

return ERROR;

} else {

List<Tickets> ticketList = new ArrayList<Tickets>();

while(rs.next()) {

ticketList.add(new Tickets(rs.getLong(0), rs.getTimestamp(1), rs.getTimestamp(2), rs.getLong(3), rs.getLong(4), rs.getInt(5),

rs.getString(6), rs.getString(7), rs.getString(8)));

}

setListOfTickets(ticketList);

return SUCCESS;

}

} catch(ClassNotFoundException e) {

System.out.println("Could not find class: " + e);

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return ERROR;

}

\*/

/\*\*

\* Use Case: Get Flight manifest.

\* Query database for a list of tickets matching the flight ID given.

\* @param flightId The flight ID of the flight for which we want the manifest.

\* @return A list of tickets for the flight ID. If no tickets are found, a null is returned.

\*/

public List<Ticket> getFlightManifest(Long flightManifest\_flightId) {

List<Ticket> ticketList = new ArrayList<Ticket>();

try{

PreparedStatement ps = null;

ResultSet rs = null;

//Class.forName(driverName).newInstance();

//con = DriverManager.getConnection(url, userName, password);

ps = con.prepareStatement("SELECT \* FROM Ticket WHERE flight\_id = ?");

ps.setLong(1, flightManifest\_flightId);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

Ticket tk = new Ticket();

tk.setId(rs.getLong("id"));

tk.setFlightId(rs.getLong("flight\_id"));

tk.setBookingId(rs.getLong("booking\_id"));

tk.setSeatType(rs.getString("seat\_type"));

tk.setSeatPref(rs.getString("seat\_pref"));

tk.setMealPref(rs.getString("meal\_pref"));

tk.setPassengerName(rs.getString("passenger\_name"));

ticketList.add(tk);

}

return ticketList;

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

public String getPlaceByCode(String code) {

try{

PreparedStatement ps = null;

ResultSet rs = null;

// get origin code

ps = con.prepareStatement("SELECT name FROM place WHERE code = ?");

ps.setString(1, code);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

String name = rs.getString("name");

return name;

}

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

/\*\*

\* Use Case: Search Flights.

\* Query database for a list of flights matching the origin/destination/requiredSeats/departureTime.

\* @param origin The origin of the flight.

\* @param destination The destination of the flight.

\* @param departureTime The expected time of departure for that flight.

\* @param requiredSeats The minimum number of seats that must be available for that flight.

\* @param isFirstClass The seat type of the flight.

\* @return A list of flights that satisfies above conditions. If no suitable flights are found, a null is returned.

\*/

public List<Flight> getFlights(String origin, String destination, Timestamp departureTime, Integer requiredSeats, boolean isFirstClass) {

List<Flight> flightList = new ArrayList<Flight>();

String origin\_code = "";

String destination\_code = "";

try{

PreparedStatement ps = null;

ResultSet rs = null;

// get origin code

ps = con.prepareStatement("SELECT code FROM place WHERE name = ?");

ps.setString(1, origin);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

origin\_code = rs.getString("code");

break;

}

}

// get destination code

ps = con.prepareStatement("SELECT code FROM place WHERE name = ?");

ps.setString(1, destination);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

destination\_code = rs.getString("code");

break;

}

}

// get time interval

Calendar cal = Calendar.getInstance();

int diff = 2;

java.util.Date ts = new java.util.Date(departureTime.getTime());

cal.setTime(ts);

cal.add(Calendar.DATE ,diff);

java.util.Date ts\_new = cal.getTime();

Timestamp departureTime\_end = new Timestamp(ts\_new.getTime());

if (isFirstClass)

ps = con.prepareStatement("SELECT \* FROM Flight WHERE origin = ? AND destination = ? AND departure\_time > ? AND departure\_time < ? AND first\_class\_avail >= ?");

else

ps = con.prepareStatement("SELECT \* FROM Flight WHERE origin = ? AND destination = ? AND departure\_time > ? AND departure\_time < ? AND economy\_avail >= ?");

ps.setString(1, origin\_code);

ps.setString(2, destination\_code);

ps.setTimestamp(3, departureTime);

ps.setTimestamp(4, departureTime\_end);

ps.setInt(5, requiredSeats);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

Flight fl = new Flight();

fl.setId(rs.getLong("id"));

fl.setOrigin(rs.getString("origin"));

fl.setOriginName(origin);

fl.setDestination(rs.getString("destination"));

fl.setDestinationName(destination);

fl.setDepartureTime(rs.getTimestamp("departure\_time"));

fl.setArrivalTime(rs.getTimestamp("arrival\_time"));

fl.setAirlineName(rs.getString("airline\_name"));

if (isFirstClass) {

fl.setAvailableSeats(rs.getInt("first\_class\_avail"));

fl.setCost(rs.getInt("base\_cost")\*5);

}

else {

fl.setAvailableSeats(rs.getInt("economy\_avail"));

fl.setCost(rs.getInt("base\_cost"));

}

flightList.add(fl);

}

return flightList;

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

public List<Flight> getFlightSelection() {

List<Flight> selectionList = new ArrayList<Flight>();

try {

PreparedStatement ps = null;

ResultSet rs = null;

//Class.forName(driverName).newInstance();

//con = DriverManager.getConnection(url, userName, password);

ps = con.prepareStatement("SELECT \* FROM flight");

rs = ps.executeQuery();

while (rs.next()) {

Flight selection = new Flight();

selection.setId(rs.getLong("id"));

selection.setOrigin(rs.getString("origin"));

selection.setDestination(rs.getString("destination"));

selection.setDepartureTime(rs.getTimestamp("departure\_time"));

selection.setArrivalTime(rs.getTimestamp("arrival\_time"));

selection.setAirlineName(rs.getString("airline\_name"));

selectionList.add(selection);

}

return selectionList;

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

public Statistics getStatistics() {

Statistics stat = new Statistics();

try {

PreparedStatement ps = null;

ResultSet rs = null;

//Count all flights that are delayed

ps = con.prepareStatement("SELECT COUNT(\*) FROM Flight WHERE arrival\_time < actual\_arrival\_time");

rs = ps.executeQuery();

rs.next();

stat.setFlightsDelayed(rs.getInt("count(\*)"));

//Count all flights that are early

ps = con.prepareStatement("SELECT COUNT(\*) FROM Flight WHERE arrival\_time > actual\_arrival\_time");

rs = ps.executeQuery();

rs.next();

stat.setFlightsEarly(rs.getInt("count(\*)"));

//Count all flights that are on time

ps = con.prepareStatement("SELECT COUNT(\*) FROM Flight WHERE arrival\_time = actual\_arrival\_time");

rs = ps.executeQuery();

rs.next();

stat.setFlightsOnTime(rs.getInt("count(\*)"));

//Count all flights that were full

ps = con.prepareStatement("SELECT COUNT(\*) FROM Flight WHERE first\_class\_avail = 0 AND economy\_avail = 0");

rs = ps.executeQuery();

rs.next();

stat.setFlightsFull(rs.getInt("count(\*)"));

//Count all flights that were less than half full

ps = con.prepareStatement("SELECT COUNT(\*) FROM Flight WHERE first\_class\_avail + economy\_avail < 50");

rs = ps.executeQuery();

rs.next();

stat.setFlightsHalfEmpty(rs.getInt("count(\*)"));

String ori;

//Count all flights heading from origin

ps = con.prepareStatement("SELECT origin, COUNT(\*) FROM Flight GROUP BY origin");

rs = ps.executeQuery();

if(rs == null) {

return null;

}

else {

Map<String, Integer> originMap = new HashMap<String, Integer>();

while (rs.next()) {

originMap.put(rs.getString("origin"), rs.getInt("count(\*)"));

/\*ori = rs.getString("origin");

ps = con.prepareStatement("SELECT name FROM place WHERE code = ?");

ps.setString(1, ori);

rs = ps.executeQuery();

rs.next();

st.setflightOriginStr(rs.getString("name"));\*/

stat.setFlightOriginMap(originMap);

}

}

String des;

//Count all flights heading to destination

ps = con.prepareStatement("SELECT destination, COUNT(\*) FROM Flight GROUP BY destination");

rs = ps.executeQuery();

if(rs == null) {

return null;

}

else {

Map<String, Integer> destinationMap = new HashMap<String, Integer>();

while (rs.next()) {

destinationMap.put(rs.getString("destination"), rs.getInt("count(\*)"));

/\*des = rs.getString("destination");

ps = con.prepareStatement("SELECT name FROM place WHERE code = ?");

ps.setString(1, des);

rs = ps.executeQuery();

rs.next();

st.setflightDestinationStr(rs.getString("name"));\*/

stat.setFlightDestinationMap(destinationMap);

}

}

String popular;

//Find most popular destination

ps = con.prepareStatement("SELECT destination, COUNT(\*) as cnt FROM Flight GROUP BY destination ORDER BY cnt desc limit 1"); //Most popular

rs = ps.executeQuery();

rs.next();

popular = rs.getString("destination");

ps = con.prepareStatement("SELECT name FROM place WHERE code = ?");

ps.setString(1, popular);

rs = ps.executeQuery();

rs.next();

stat.setMostPopular(rs.getString("name"));

return stat;

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

public String getMessage() {

return message;

}

public void setMessage(String message) {

this.message = message;

}

public String getDestination() {

return destination;

}

public void setDestination(String destination) {

this.destination = destination;

}

public String getDestination2() {

return destination2;

}

public void setDestination2(String destination2) {

this.destination2 = destination2;

}

public String getFlightId() {

return flightId;

}

public void setFlightId(String flightId) {

this.flightId = flightId;

}

public String getFlightId2() {

return flightId2;

}

public void setFlightId2(String flightId2) {

this.flightId2 = flightId2;

}

public String getOrigin() {

return origin;

}

public void setOrigin(String origin) {

this.origin = origin;

}

public String getDepartureTime() {

return departureTime;

}

public void setDepartureTime(String departureTime) {

this.departureTime = departureTime;

}

public String getAvailableSeats() {

return availableSeats;

}

public void setAvailableSeats(String availableSeats) {

this.availableSeats = availableSeats;

}

public List<Flight> getListOfFlights() {

return listOfFlights;

}

public void setListOfFlights(List<Flight> listOfFlights) {

this.listOfFlights = listOfFlights;

}

public User getUser() {

return user;

}

public void setUser(User user) {

this.user = user;

}

}

FlightServiceTest.java

package ubc.ece419.pod10;

import java.util.List;

import junit.framework.TestCase;

import ubc.ece419.pod10.action.user.UserSearchFlightsAction;

import ubc.ece419.pod10.domain.Flight;

import ubc.ece419.pod10.service.FlightService;

import com.opensymphony.xwork2.ActionSupport;

public class FlightServiceTest extends TestCase {

public static List<Flight> listOfFlights;

private static boolean testGetFlights = false;

FlightService flightService = new FlightService();

public void testGetFlights() throws Exception

{

try {

listOfFlights = flightService.getFlightSelection();

} catch (Exception e) {

//continueTest = false;

System.out.println(e.getMessage());

fail("==TEST ABORTED: Get Flight Lists==");

return;

}

if(listOfFlights == null)

{

fail("Null Flight Lists returned");

}

testGetFlights = true;

}

public void testSearchFlights() throws Exception

{

if(!testGetFlights)

{

testGetFlights();

}

List<Flight> searchResult;

boolean found = false;

String msg;

for(int i=0; i<listOfFlights.size(); i++)

{

Flight fl = new Flight();

fl = listOfFlights.get(i);

try{

searchResult = flightService.getFlights(fl.getOrigin(), fl.getDestination(), fl.getDepartureTime(), 1, false);

found = false;

for(int j=0; j<searchResult.size();j++)

{

Flight flret = new Flight();

flret = searchResult.get(j);

if(fl.getId() == flret.getId())

{

found = true;

break;

}

msg = "Cannot find ID " + fl.getId() + " from " + fl.getOrigin() + " to " + fl.getDestination();

assertTrue("msg", found);

}

} catch (Exception e) {

System.out.println("flightService.getFlights failed");

System.out.println(e.getMessage());

}

}

}

}

InvalidBoardingPassException.java

package ubc.ece419.pod10.exceptions;

public class InvalidBoardingPassException extends Exception{

public InvalidBoardingPassException() {

super();

}

public InvalidBoardingPassException(String s) {

super(s);

}

public InvalidBoardingPassException(Throwable t, String s) {

super(s, t);

}

public InvalidBoardingPassException(Throwable t) {

super(t);

}

}

InvalidBookingException.java

package ubc.ece419.pod10.exceptions;

public class InvalidBookingException extends Exception{

public InvalidBookingException() {

super();

}

public InvalidBookingException(String s) {

super(s);

}

public InvalidBookingException(Throwable t, String s) {

super(s, t);

}

public InvalidBookingException(Throwable t) {

super(t);

}

}

InvalidTicketException.java

package ubc.ece419.pod10.exceptions;

public class InvalidTicketException extends Exception{

public InvalidTicketException() {

super();

}

public InvalidTicketException(String s) {

super(s);

}

public InvalidTicketException(Throwable t, String s) {

super(s, t);

}

public InvalidTicketException(Throwable t) {

super(t);

}

}

InvalidTicketModificationException.java

package ubc.ece419.pod10.exceptions;

public class InvalidTicketModificationException extends Exception{

public InvalidTicketModificationException() {

super();

}

public InvalidTicketModificationException(String s) {

super(s);

}

public InvalidTicketModificationException(Throwable t, String s) {

super(s, t);

}

public InvalidTicketModificationException(Throwable t) {

super(t);

}

}

LoginAction.java

package ubc.ece419.pod10.action.user;

import java.math.BigInteger;

import java.sql.Timestamp;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.service.UserService;

public class LoginAction extends BaseAction {

UserService userService = new UserService();

String password, emailAddress, errorMessage;

Boolean loggedIn = false;

String userId;

public String execute() {

// If both emailAddress and password are null, user just landed on page

if (password == null && emailAddress == null) {

return INPUT;

}

if (password == null || emailAddress == null) {

errorMessage = "Please fill out all the fields";

System.out.println("Please fill out all the fields");

return ERROR;

}

Long userIdFromDb = userService.loginCheck(emailAddress, password);

if (userIdFromDb == null) {

errorMessage = "Incorrect password for input email.";

System.out.println("Incorrect password for input email.");

return ERROR;

}

userId = userIdFromDb.toString();

user = userService.getUserById(Long.parseLong(userId));

if (user.getUserType() != 0) {

return ERROR;

}

// Generate session token and store it in the user table.

try {

userService.updateSessionToken(userIdFromDb);

} catch(Exception e) {

System.out.println("Could not update session");

}

// Redirect to search page.

loggedIn = true;

return SUCCESS;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getEmailAddress() {

return emailAddress;

}

public void setEmailAddress(String emailAddress) {

this.emailAddress = emailAddress;

}

public String getErrorMessage() {

return errorMessage;

}

public void setErrorMessage(String errorMessage) {

this.errorMessage = errorMessage;

}

public Boolean getLoggedIn() {

return loggedIn;

}

public void setLoggedIn(Boolean loggedIn) {

this.loggedIn = loggedIn;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

}

LoginErrorException.java

package ubc.ece419.pod10.exceptions;

public class LoginErrorException extends Exception {

public LoginErrorException() {

super();

}

public LoginErrorException(String s) {

super(s);

}

public LoginErrorException(Throwable t, String s) {

super(s, t);

}

public LoginErrorException(Throwable t) {

super(t);

}

}

ModifyTicketAction.java

package ubc.ece419.pod10.action.user;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Ticket;

import ubc.ece419.pod10.exceptions.InvalidTicketModificationException;

import ubc.ece419.pod10.service.ReservationService;

import com.opensymphony.xwork2.ActionSupport;

public class ModifyTicketAction extends BaseAction {

String ticketId, seatPreference, mealPreference, passengerName, userId;

Ticket ticket;

public Ticket getTicket() {

return ticket;

}

public void setTicket(Ticket ticket) {

this.ticket = ticket;

}

ReservationService reservationService = new ReservationService();

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if(ticketId == null || seatPreference == null || mealPreference == null || passengerName == null) {

return ERROR;

}

Long ticketIdInt = Long.parseLong(ticketId);

try {

ticket = reservationService.modifyTicket(ticketIdInt, seatPreference, mealPreference, passengerName);

} catch (InvalidTicketModificationException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return SUCCESS;

}

public String getModifyTicketGuiInput(){

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (ticketId == null){

return ERROR;

}

return SUCCESS;

}

public String getSeatPreference() {

return seatPreference;

}

public void setSeatPreference(String seatPreference) {

this.seatPreference = seatPreference;

}

public String getMealPreference() {

return mealPreference;

}

public void setMealPreference(String mealPreference) {

this.mealPreference = mealPreference;

}

public String getPassengerName() {

return passengerName;

}

public void setPassengerName(String passengerName) {

this.passengerName = passengerName;

}

public String getticketId() {

return ticketId;

}

public void setticketId(String ticketId) {

this.ticketId = ticketId;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

}

Points.java

package ubc.ece419.pod10.domain;

public class Points {

}

ReservationService.java

package ubc.ece419.pod10.service;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.domain.Billing;

import ubc.ece419.pod10.domain.BoardingPass;

import ubc.ece419.pod10.domain.Booking;

import ubc.ece419.pod10.domain.Flight;

import ubc.ece419.pod10.domain.Ticket;

import ubc.ece419.pod10.domain.TicketSet;

import ubc.ece419.pod10.domain.User;

import ubc.ece419.pod10.exceptions.InvalidBoardingPassException;

import ubc.ece419.pod10.exceptions.InvalidBookingException;

import ubc.ece419.pod10.exceptions.InvalidTicketException;

import ubc.ece419.pod10.exceptions.InvalidTicketModificationException;

import ubc.ece419.pod10.exceptions.ViewedTooEarlyException;

import com.opensymphony.xwork2.ActionSupport;

public class ReservationService extends ActionSupport {

private static final long serialVersionUID = 1L;

private static final String URL = "jdbc:mysql://localhost:3306/pod10\_db?zeroDateTimeBehavior=convertToNull";

private static final String DRIVER\_NAME = "com.mysql.jdbc.Driver";

private static final String DB\_USER\_NAME = "root";

private static final String DB\_PASSWORD = "ECE419\_pod10";

Connection con = null;

Statement stmt = null;

public ReservationService() {

super();

try {

Class.forName(DRIVER\_NAME).newInstance();

con = DriverManager.getConnection(URL, DB\_USER\_NAME, DB\_PASSWORD);

} catch(InstantiationException e) {

e.printStackTrace();

} catch(IllegalAccessException e) {

e.printStackTrace();

} catch(ClassNotFoundException e) {

e.printStackTrace();

} catch (SQLException e) {

e.printStackTrace();

}

}

/\*\*

\* Use Case: Reserve Flights

\* add a booking entry in Booking Table to associate all tickets booked with this userId

\* @param userID used to associate with the generated bookingId

\* @return generated bookingId

\*/

public Long addBooking(Long userId) {

PreparedStatement ps = null;

ResultSet rs = null;

int cancelState = 1;//allowed to cancel

Long bookingId = null;

try {

ps = con.prepareStatement("INSERT INTO booking(date\_created, date\_modified, user\_id, cancel\_state) VALUES(now(), now(), ?,?)",Statement.RETURN\_GENERATED\_KEYS);

ps.setLong(1, userId);

ps.setInt(2, cancelState);

int val = ps.executeUpdate();

if (val == 0) {

return null;

} else {

rs = ps.getGeneratedKeys();

if (rs == null) {

return null;

} else {

while (rs.next()) {

bookingId = rs.getLong(1);

}

return bookingId;

}

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

/\*\*

\* Use Case: Reserve Flights

\* Add a new ticket instance into Ticket Table

\* @param ps the object holds and precompiles the SQL statement

\* @param bookingId has the ID of the lasted booking

\* @param flightId has the ID of the flight associated with this ticket

\* @param bookingId all booked tickets of a single userId are associated with this bookingId

\* @param passengerName is the name of the person who uses this ticket

\* @param seatType is user's choice for the seat type of this ticket

\* @param seatPref is user's choice for the seat class of this ticket

\* @param mealPref is user's choice for the meal of this ticket

\* @return a Flight object with all tickets and information of each ticket booked with this userId

\*/

public List<Flight> addTickets(Long userId, Long flightId, Long bookingId, String passengerName, String seatType, String seatPref, String mealPref){

PreparedStatement ps = null;

ResultSet rs = null;

List<Flight> flightList = new ArrayList<Flight>();

int state = 0;

int cost = 0;

try {

// get flight cost

ps = con.prepareStatement("SELECT base\_cost FROM flight WHERE id = ?");

ps.setLong(1, flightId);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

cost = rs.getInt("base\_cost");

if (cost == 0) {

return null;

}

}

}

if (seatType.equals("1"))

cost = cost \* 5;

// add entry to ticket table

if (bookingId == null) {

return null;

} else {

int val = addTicket(ps,bookingId, flightId, passengerName, seatType,seatPref, mealPref, state, cost);

if (val == 0) {

return null;

} else {

ps = con.prepareStatement("SELECT \* FROM Flight WHERE id = ?");

ps.setLong(1, flightId);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

Flight fl = new Flight();

fl.setId(rs.getLong("id"));

fl.setOrigin(rs.getString("origin"));

fl.setOriginName(getPlaceByCode(fl.getOrigin()));

fl.setDestination(rs.getString("destination"));

fl.setDestinationName(getPlaceByCode(fl.getDestination()));

fl.setDepartureTime(rs.getTimestamp("departure\_time"));

fl.setArrivalTime(rs.getTimestamp("arrival\_time"));

fl.setAirlineName(rs.getString("airline\_name"));

if (seatType.equals("1"))

fl.setCost(rs.getInt("base\_cost")\*5);

else

fl.setCost(rs.getInt("base\_cost"));

flightList.add(fl);

}

return flightList;

}

}

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

public String getPlaceByCode(String code) {

try{

PreparedStatement ps = null;

ResultSet rs = null;

// get origin code

ps = con.prepareStatement("SELECT name FROM place WHERE code = ?");

ps.setString(1, code);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

String name = rs.getString("name");

return name;

}

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

/\*\*

\* Use Case: Reserve Flights

\* Add a new ticket instance into Ticket Table

\* @param ps the object holds and precompiles the SQL statement

\* @param bookingId has the ID of the lasted booking

\* @param flightId has the ID of the flight associated with this ticket

\* @param passengerName is the name of the person who uses this ticket

\* @param seatType is user's choice for the seat type of this ticket

\* @param seatPref is user's choice for the seat class of this ticket

\* @param mealPref is user's choice for the meal of this ticket

\* @param state indicates if the ticked is reserved, paid, or unpaid

\* @param cost is the charge of this ticket

\* @return an int result indicates if update ticket table is successful, 0=failed, otherwise= success

\*/

public int addTicket(PreparedStatement ps, Long bookingId, Long flightId, String passengerName, String seatType, String seatPref, String mealPref, int state, int cost){

try {

ps = con.prepareStatement("INSERT INTO ticket (flight\_id, booking\_id, date\_created, date\_modified, seat\_type, seat\_pref, meal\_pref, passenger\_name, cost, state) VALUES(?, ?, now(), now(), ?, ?,?,?,?,?)");

ps.setLong(1, flightId);

ps.setLong(2, bookingId);

ps.setString(3, seatType);

ps.setString(4, seatPref);

ps.setString(5, mealPref);

ps.setString(6, passengerName);

ps.setInt(7, cost);

ps.setInt(8, state);

int val = ps.executeUpdate();

return val;

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return 0;

}

/\*\*

\* Use Case: Cancel Booking

\* Cancels a Booking based on booking Id

\* @param bookingId has the ID of the booking to cancel

\* @return an int result indicates if the booking cancel is successful, 1=success, otherwise= failed

\*/

public int cancelBooking(Long bookingId) {

PreparedStatement ps = null;

try {

ps = con.prepareStatement("UPDATE booking SET cancel\_state = 2, date\_modified = now() WHERE id = ?");

ps.setLong(1, bookingId);

int val = ps.executeUpdate();

if (val != 1) {

// To few or too many records

throw new InvalidBookingException("This booking is invalid. Contact an agent at 1-800-POD-2020");

}

return val;

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return 0;

}

/\*\*

\* Use Case: Cancel Booking

\* Cancels all tickets associated with the booking with booking Id

\* @param bookingId has the ID of the booking for which all tickets shall be cancelled

\* @return an int result indicates how many tickets have been cancelled, 0=failed, 1 or more= success

\*/

public int cancelTickets(Long bookingId) {

PreparedStatement ps = null;

try {

ps = con.prepareStatement("UPDATE ticket SET state = 3, date\_modified = now() WHERE booking\_id = ?");

ps.setLong(1, bookingId);

int val = ps.executeUpdate();

if (val < 1) {

// To few or too many records

throw new InvalidBookingException("This booking is invalid. Contact an agent at 1-800-POD-2020");

}

return val;

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return 0;

}

/\*\*

\* Use Case: Cancel Booking

\* Refunds a bill based on a billing Id

\* @param billing has the ID of the bill to refund

\* @return an int result indicates if the billing refund is successful, 1=success, otherwise= failed

\*/

public int refundBill(Long billingId) {

Billing bill = getBill(billingId);

Integer refundAmount = (int) (-0.7 \* bill.getAmount());

int var;

try {

ArrayList<Long> res = addBill(bill, refundAmount);

var = res.get(0).intValue();

if (bill.isLoyaltyPointsUsed()){

// Add a new points entry if this was a points payment (to refund the points paid)

//refundPoints(bill.getUserId(), bill.getBookingId(), refundAmount);

updateUserPoints(bill.getUserId(), refundAmount, res.get(1), "Refund Points");

}

return var;

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return 0;

}

public Billing getBill(Long billId){

PreparedStatement ps = null;

ResultSet rs = null;

Billing bill = new Billing();

try {

ps = con.prepareStatement("SELECT id, user\_id, booking\_id, date\_created, amount, name, credit\_card\_number, is\_loyalty\_points\_used FROM billing WHERE id = ?");

ps.setLong(1, billId);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

bill.setId(rs.getLong("id"));

bill.setUserId(rs.getLong("user\_id"));

bill.setBookingId(rs.getLong("booking\_id"));

bill.setDateCreated(rs.getTimestamp("date\_created"));

bill.setAmount(rs.getInt("amount"));

bill.setLoyaltyPointsUsed(rs.getBoolean("is\_loyalty\_points\_used"));

if (!bill.getIsLoyaltyPointsUsed()) {

bill.setName(rs.getString("name"));

bill.setCreditCardNum(rs.getString("credit\_card\_number"));

}

}

return bill;

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

public ArrayList<Long> addBill(Billing bill, Integer refund){

ArrayList<Long> resList = new ArrayList<Long>();

PreparedStatement ps = null;

ResultSet rs = null;

try {

ps = con.prepareStatement("INSERT INTO billing (date\_created, date\_modified, user\_id, booking\_id, amount, name, credit\_card\_number, is\_loyalty\_points\_used) VALUES(now(), now(), ?, ?, ?, ?, ?, ?)",Statement.RETURN\_GENERATED\_KEYS);

ps.setLong(1, bill.getUserId());

ps.setLong(2, bill.getBookingId());

ps.setInt(3, refund);

if (bill.getIsLoyaltyPointsUsed()){

ps.setString(4, null);

ps.setString(5, null);

} else {

ps.setString(4, bill.getName());

ps.setString(5, bill.getCreditCardNum());

}

ps.setBoolean(6, bill.getIsLoyaltyPointsUsed());

int val = ps.executeUpdate();

resList.add((long)val);

if (val != 1) {

// To few or too many records

throw new InvalidBookingException("This booking is invalid. Contact an agent at 1-800-POD-2020");

}

else if (val == 1)

{

Long billingId = null;

rs = ps.getGeneratedKeys();

if (rs == null) {

return null;

} else {

while (rs.next()) {

billingId = rs.getLong(1);

break;

}

resList.add(billingId);

}

}

return resList;

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

private int refundPoints(Long userId, Long billingId, Integer amount){

PreparedStatement ps = null;

try {

// By using LIMIT 1, we only add one points set into the database even if there are 2 billing records

ps = con.prepareStatement("INSERT INTO points (date\_created, date\_modified, user\_id, billing\_id, amount) VALUES(now(), now(), ?, ?, ?)",Statement.RETURN\_GENERATED\_KEYS);

ps.setLong(1, userId);

ps.setLong(2, billingId);

ps.setInt(3, amount);

int val = ps.executeUpdate();

if (val != 1) {

// To few or too many records

throw new InvalidBoardingPassException("No billing information is available for this boarding pass. Contact an agent at 1-800-POD-1013");

}

return val;

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return 0;

}

/\*\*

\* Use Case: reserve Flights

\* check if users can make payment with their loyal point

\* @param userId current logged in account userId

\* @param bookingId associated with current account balance

\* @return returns a boolean value indicates if users can pay with loyal points

\*/

public boolean canUsePoints(Long userId, Long bookingId) {

int currBalance = getCurrBalance(userId);

int actualCost = getBookingSubtotal(bookingId);

return actualCost <= currBalance;

}

public int getPointsBalance(Long userId) {

return getCurrBalance(userId);

}

/\*\*

\* Use Case: Manage Payment

\* get current account loyalty points balance

\* @param userId current logged in account userId

\* @return returns an integer type variable of current account loyalty points balance

\*/

public int getBookingCancelState(Long bookingId) {

PreparedStatement ps = null;

ResultSet rs = null;

int cancelState = -1;

try {

ps = con.prepareStatement("SELECT cancel\_state FROM booking WHERE id = ?");

ps.setLong(1, bookingId);

rs = ps.executeQuery();

if (rs != null) {

while (rs.next()) {

cancelState = rs.getInt("cancel\_state");

}

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return cancelState;

}

private int getCurrBalance(Long userId){

PreparedStatement ps = null;

ResultSet rs = null;

int currBalance=0;

try {

// check if user have enough points to pay for the bill

ps = con.prepareStatement("SELECT current\_point\_balance FROM user WHERE id = ?");

ps.setLong(1, userId);

rs = ps.executeQuery();

if (rs != null) {

while (rs.next()) {

currBalance = rs.getInt("current\_point\_balance");

}

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return currBalance;

}

/\*\*

\* Use Case: reserve Flights

\* get current account balance

\* @param userId current logged in account userId

\* @return returns an integer type variable of current account balance

\*/

public Integer getBookingSubtotal(Long bookingId) {

PreparedStatement ps = null;

ResultSet rs = null;

Integer total = null;

try {

ps = con.prepareStatement("SELECT sum(cost) as total FROM ticket WHERE booking\_id = ?");

ps.setLong(1, bookingId);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

total = rs.getInt("total");

}

return total;

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

public Booking getCurrentBooking(Long bookingId) {

Booking bk = new Booking();

try {

PreparedStatement ps = con.prepareStatement("SELECT id, date\_created, date\_modified, user\_id, cancel\_state FROM booking WHERE id=?");

ps.setLong(1, bookingId);

ResultSet rs = ps.executeQuery();

while (rs.next()) {

bk.setId(rs.getLong("id"));

bk.setDateCreated(rs.getTimestamp("date\_created"));

bk.setDateModified(rs.getTimestamp("date\_modified"));

bk.setUserID(rs.getLong("user\_id"));

bk.setCancelState(rs.getLong("cancel\_state"));

}

return bk;

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

public List<Booking> getBookingList(Long userId){

PreparedStatement ps = null;

ResultSet rs = null;

List<Booking> bookingList = new ArrayList<Booking>();

try {

ps = con.prepareStatement("SELECT id, date\_created, date\_modified, user\_id, cancel\_state FROM booking WHERE user\_id = ?");

ps.setLong(1, userId);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

Booking bk = new Booking();

bk.setId(rs.getLong("id"));

bk.setDateCreated(rs.getTimestamp("date\_created"));

bk.setDateModified(rs.getTimestamp("date\_modified"));

bk.setUserID(rs.getLong("user\_id"));

bk.setCancelState(rs.getLong("cancel\_state"));

bookingList.add(bk);

}

return bookingList;

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

public Billing getCurrentBilling(Long bookingId){

PreparedStatement ps = null;

ResultSet rs = null;

Billing bill = new Billing();

try {

ps = con.prepareStatement("SELECT id, booking\_id, date\_created, amount, name, credit\_card\_number, is\_loyalty\_points\_used FROM billing WHERE booking\_id = ?");

ps.setLong(1, bookingId);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

bill.setId(rs.getLong("id"));

bill.setBookingId(rs.getLong("booking\_id"));

bill.setDateCreated(rs.getTimestamp("date\_created"));

bill.setAmount(rs.getInt("amount"));

bill.setLoyaltyPointsUsed(rs.getBoolean("is\_loyalty\_points\_used"));

if (!bill.getIsLoyaltyPointsUsed()) {

bill.setName(rs.getString("name"));

bill.setCreditCardNum(rs.getString("credit\_card\_number"));

}

}

return bill;

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

/\*\*

\* Use Case: reserve flights

\* get all tickets associated with the bookingId

\* @param bookingId used to get all associated tickets

\* @return a list of ticket objects

\*/

public List<Ticket> getCurrentTickets(Long bookingId) {

PreparedStatement ps = null;

ResultSet rs = null;

List<Ticket> ticketList = new ArrayList<Ticket>();

try {

ps = con.prepareStatement("SELECT \* FROM ticket WHERE booking\_id = ?");

ps.setLong(1, bookingId);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

Ticket t = new Ticket();

t.setId(rs.getLong("id"));

t.setFlightId(rs.getLong("flight\_id"));

t.setSeatType(rs.getString("seat\_type"));

t.setSeatPref(rs.getString("seat\_pref"));

t.setMealPref(rs.getString("meal\_pref"));

t.setPassengerName(rs.getString("passenger\_name"));

t.setCost(rs.getLong("cost"));

t.setState(rs.getLong("state"));

ticketList.add(t);

}

return ticketList;

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

public List<TicketSet> getTicketSetList(Long bookingId) {

PreparedStatement ps = null;

ResultSet rs = null;

List<TicketSet> ticketList = new ArrayList<TicketSet>();

try {

ps = con.prepareStatement("SELECT t.\*, f.origin, f.destination, f.departure\_time, f.arrival\_time, f.airline\_name FROM ticket t, flight f WHERE t.booking\_id = ? AND f.id = t.flight\_id");

ps.setLong(1, bookingId);

rs = ps.executeQuery();

if (rs == null) {

return null;

} else {

while (rs.next()) {

Ticket t = new Ticket();

Flight fl = new Flight();

TicketSet ticketSet = new TicketSet();

t.setId(rs.getLong("id"));

t.setFlightId(rs.getLong("flight\_id"));

t.setSeatType(rs.getString("seat\_type"));

t.setSeatPref(rs.getString("seat\_pref"));

t.setMealPref(rs.getString("meal\_pref"));

t.setPassengerName(rs.getString("passenger\_name"));

t.setCost(rs.getLong("cost"));

t.setState(rs.getLong("state"));

fl.setOrigin(rs.getString("origin"));

fl.setDestination(rs.getString("destination"));

fl.setDepartureTime(rs.getTimestamp("departure\_time"));

fl.setArrivalTime(rs.getTimestamp("arrival\_time"));

fl.setAirlineName(rs.getString("airline\_name"));

ticketSet.setTicket(t);

ticketSet.setFlight(fl);

ticketList.add(ticketSet);

}

return ticketList;

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

/\*\*

\* Use Case: reserve Flights

\* add a new entry into Billing Table

\* @param userId current logged in account id

\* @param bookingId the bookingId that associates with the retrieved billing information

\* @param amount the cost of this billing entry

\* @param cardHolder name of the credit card holder

\* @param cardNum number of the credit card

\* @param isPaidByPoints indicates if this bill is already paid by using loyal points

\* @return a Billing object contains information of this billing

\*/

public Billing addBilling(Long userId, Long bookingId, int amount, String cardHolder, String cardNum, boolean isPaidByPoints){

PreparedStatement ps = null;

ResultSet rs = null;

Billing b = null;

Long billingId = null;

try {

ps = con.prepareStatement("INSERT INTO billing(date\_created, date\_modified, user\_id, booking\_id, amount, name, credit\_card\_number, is\_loyalty\_points\_used) VALUES (now(), now(), ?,?,?,?,?,?)",Statement.RETURN\_GENERATED\_KEYS);

ps.setLong(1, userId);

ps.setLong(2, bookingId);

ps.setInt(3, amount);

ps.setString(4, cardHolder);

ps.setString(5, cardNum);

ps.setBoolean(6, isPaidByPoints);

int val = ps.executeUpdate();

if (val == 0) {

return null;

} else {

rs = ps.getGeneratedKeys();

if (rs == null) {

return null;

} else {

while (rs.next()) {

billingId = rs.getLong(1);

b = new Billing();

b.setId(billingId);

b.setAmount(amount);

b.setBookingId(bookingId);

b.setLoyaltyPointsUsed(isPaidByPoints);

if (!isPaidByPoints) {

b.setName(cardHolder);

b.setCreditCardNum(cardNum);

}

break;

}

updateTicketState(bookingId);

updateFlightSeat(bookingId);

if (isPaidByPoints) {

updateUserPoints(userId, amount, billingId,"Pay by Points");

}

return b;

}

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

/\*\*

\* Use Case: reserve Flights

\* update current user's loyalty points

\* @param userId current account Id

\* @param amount balance of the paid ticket

\*/

private void updateUserPoints(Long userId, int amount, Long billingId, String reason) {

PreparedStatement ps = null;

int res = 0;

try {

ps = con.prepareStatement("UPDATE user SET current\_point\_balance = current\_point\_balance-?, date\_modified = now() WHERE id=?");

ps.setInt(1, amount);

ps.setLong(2, userId);

res = ps.executeUpdate();

if (res != 1){

throw new InvalidBoardingPassException("Points update was not successful. Contact an agent at 1-800-POD-1013");

}

// in points table

ps = con.prepareStatement("INSERT INTO points (date\_created,date\_modified,user\_id,billing\_id,amount,reason) VALUES (now(), now(), ?, ?, ?,?)");

ps.setLong(1, userId);

ps.setLong(2, billingId);

ps.setInt(3, -amount);

ps.setString(4, reason);

res = ps.executeUpdate();

if(res != 1) {

throw new InvalidBoardingPassException("Points update was not successful. Contact an agent at 1-800-POD-1013");

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

}

/\*\*

\* Use Case: reserve Flights

\* save user choice of seat type

\* @param bookingId the updated ticket's bookingId

\*/

private void updateFlightSeat(Long bookingId) {

PreparedStatement ps = null;

ResultSet rs = null;

try {

ps = con.prepareStatement("SELECT flight\_id,seat\_type FROM ticket WHERE booking\_id = ?");

ps.setLong(1, bookingId);

rs = ps.executeQuery();

if (rs == null) {

// throw exception

} else {

while (rs.next()) {

Long flightId = rs.getLong("flight\_id");

String seatType = rs.getString("seat\_type");

PreparedStatement ps1 = null;

if (seatType.equals("1"))

ps1 = con.prepareStatement("UPDATE flight SET first\_class\_avail = first\_class\_avail-1, date\_modified = now() WHERE id = ?");

else

ps1 = con.prepareStatement("UPDATE flight SET economy\_avail = economy\_avail-1, date\_modified = now() WHERE id = ?");

ps1.setLong(1, flightId);

int res = ps1.executeUpdate();

if (res != 1){

// throw exception here

}

}

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

}

/\*\*

\* Use Case: reserve Flights

\* update ticket state if ticket if reserved or paid

\* @param bookingId the updated ticket's bookingId

\*/

private void updateTicketState(Long bookingId) {

PreparedStatement ps = null;

int res = 0;

try {

ps = con.prepareStatement("UPDATE ticket SET state = 1, date\_modified = now() WHERE booking\_id = ?");

ps.setLong(1, bookingId);

res = ps.executeUpdate();

if (res != 1){

// throw exception here

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

}

/\*\*

\* updates bookings state from previousState to newState

\* @param ticketId Identifies the ticket

\* @param newState Identifies the new state to be updated to

\* @param previousState Identifies the previous state to be update from.

\*/

private void updateTicketState(Long ticketId, Long newState, Long previousState) throws InvalidTicketException {

try {

PreparedStatement ps = con.prepareStatement("UPDATE ticket SET state = ?, date\_modified = now() WHERE id = ? AND state = ?");

ps.setLong(1, newState);

ps.setLong(2, ticketId);

ps.setLong(3, previousState);

Integer result = ps.executeUpdate();

if (!result.equals(1)) {

throw new InvalidTicketException("Could not update ticket with new state for ticket id " + ticketId);

}

} catch(SQLException e) {

throw new InvalidTicketException("Could not update ticket with new state for ticket id " + ticketId);

}

}

/\*\*

\* Updates booking cancel state from previousState to newState

\* @param bookingId Identifies the booking

\* @param newState Identifies the new state to be updated to

\* @param previousState Identifies the previous state to be updated from

\* @throws InvalidBookingException when booking cannot be updated.

\*/

//updateBookingCancelState(bookingId, 0L, 1L);

private void updateBookingCancelState(Long bookingId, Long newState, Long previousState) throws InvalidBookingException{

try {

PreparedStatement ps = con.prepareStatement("UPDATE booking SET cancel\_state=?, date\_modified = now() where id = ? AND cancel\_state = ?");

ps.setLong(1, newState);

ps.setLong(2, bookingId);

ps.setLong(3, previousState);

Integer result = ps.executeUpdate();

if (!result.equals(1)) {

throw new InvalidBookingException ("Could not update booking's cancel state for booking Id " + bookingId);

}

} catch(SQLException e) {

throw new InvalidBookingException ("Could not update booking's cancel state for booking Id " + bookingId);

}

}

/\*\*

\* Use Case: Get Boarding Pass

\* Generate a printable boarding pass for a given Ticket, showing information from the Flight and Booking

\* @param bookingId The ID of the booking for which we want to generate boarding passes

\* @return A boarding pass object, with Ticket, Booking, and Flight information

\*/

public List<BoardingPass> getBoardingPass(Long bookingId) throws ViewedTooEarlyException, InvalidTicketException, InvalidBookingException, InvalidBoardingPassException{

List<BoardingPass> boardingPassList = new ArrayList<BoardingPass>();

// 1. Get a list of tickets for that booking Id.

// 2. Print boarding passes for each ticket.

// 3. If the state of the ticket is 1, update it to 0.

// 4. If the booking's is cancelState is 1 (cancel allowed), update it to 1 and award any relevant points.

Booking booking = getBookingById(bookingId);

List<Ticket> ticketList = getTicketsForBookingId(bookingId);

for (Ticket ticket: ticketList) {

BoardingPass boardingPass = generateBoardingPass(ticket.getId());

if(ticket.getState() == 1L) {

// Update ticket state to 2.

updateTicketState(ticket.getId(), 0L, 1L);

}

boardingPassList.add(boardingPass);

}

if (booking.getCancelState() == 1L) {

// If booking = 1 meaning we can cancel it, set cancel state to 0 so we cannot cancel it after printing boarding pass

// Update booking's cancel state to 0.

// Award points if necessary.

updateBookingCancelState(bookingId, 0L, 1L);

//private void updateUserPoints(Long userId, int amount, Long billingId, String reason)

int billingAmount = getBillingAmount(bookingId);

Long billingId = getBillingId(bookingId);

updateUserPoints(booking.getUserID(),-billingAmount,billingId,"Points Reward");

}

// Display the boarding pass

return boardingPassList;

}

private int getBillingAmount(Long bookingId) {

// TODO Auto-generated method stub

int billingAmount = 0;

try {

PreparedStatement ps = con.prepareStatement("SELECT amount FROM billing WHERE booking\_id = ?");

ps.setLong(1, bookingId);

ResultSet rs = ps.executeQuery();

if (rs != null) {

while (rs.next()) {

billingAmount = rs.getInt("amount");

}

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return billingAmount;

}

private Long getBillingId(Long bookingId) {

// TODO Auto-generated method stub

Long billingId = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT id FROM billing WHERE booking\_id = ?");

ps.setLong(1, bookingId);

ResultSet rs = ps.executeQuery();

if (rs != null) {

while (rs.next()) {

billingId = rs.getLong("id");

return billingId;

}

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return billingId;

}

private List<Ticket> getTicketsForBookingId(Long bookingId) throws InvalidBookingException {

List<Ticket> ticketList = new ArrayList<Ticket>();

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM ticket WHERE booking\_id = ?");

ps.setLong(1, bookingId);

ResultSet rs = ps.executeQuery();

while(rs.next()){

Ticket ticket = new Ticket();

ticket.setId(rs.getLong("id"));

ticket.setDateCreated(rs.getTimestamp("date\_created"));

ticket.setDateModified(rs.getTimestamp("date\_modified"));

ticket.setFlightId(rs.getLong("flight\_id"));

ticket.setBookingId(rs.getLong("booking\_id"));

ticket.setSeatType(rs.getString("seat\_type"));

ticket.setSeatPref(rs.getString("seat\_pref"));

ticket.setMealPref(rs.getString("meal\_pref"));

ticket.setPassengerName(rs.getString("passenger\_name"));

ticket.setCost(rs.getLong("cost"));

ticket.setState(rs.getLong("state"));

ticketList.add(ticket);

}

return ticketList;

} catch(SQLException e) {

throw new InvalidBookingException("Could not fetch tickets for Booking ID" + bookingId);

}

}

private Booking getBookingById(Long bookingId) throws InvalidBookingException {

Booking booking = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM booking WHERE id=?");

ps.setLong(1, bookingId);

ResultSet rs = ps.executeQuery();

while (rs.next()) {

booking = new Booking();

booking.setId(rs.getLong("id"));

booking.setDateCreated(rs.getTimestamp("date\_created"));

booking.setDateModified(rs.getTimestamp("date\_modified"));

booking.setUserID(rs.getLong("user\_id"));

booking.setCancelState(rs.getLong("cancel\_state"));

}

return booking;

} catch(SQLException e) {

throw new InvalidBookingException("Could not fetch tickets for Booking Id" + bookingId);

}

}

// Generate a boarding Pass

private BoardingPass generateBoardingPass(Long ticketId) throws InvalidTicketException {

PreparedStatement ps = null;

ResultSet rs = null;

try {

// get user name

ps = con.prepareStatement("SELECT t.id, t.seat\_type, t.seat\_pref, t.meal\_pref, t.passenger\_name, b.id, f.id, f.origin, f.destination, f.departure\_time, f.arrival\_time, f.airline\_name FROM ticket t, booking b, flight f WHERE t.id = ? AND b.id = t.booking\_id AND f.id = t.flight\_id");

ps.setLong(1, ticketId);

rs = ps.executeQuery();

rs.beforeFirst();

rs.last();

// Check to see if rs size is 0

if (rs.getRow() == 0) {

// We didn't return any data, our ticketId was invalid

throw new InvalidTicketException("Ticket ID was invalid.");

}

rs.beforeFirst();

while (rs.next()) {

Booking bk = new Booking();

Flight fl = new Flight();

Ticket tk = new Ticket();

// The following sets take the args in order from the SELECT statement above, order cannot be changed

tk.setId(rs.getLong(1));

tk.setSeatType(rs.getString(2));

tk.setSeatPref(rs.getString(3));

tk.setMealPref(rs.getString(4));

tk.setPassengerName(rs.getString(5));

bk.setId(rs.getLong(6));

fl.setId(rs.getLong(7));

fl.setOrigin(rs.getString(8));

fl.setDestination(rs.getString(9));

fl.setDepartureTime(rs.getTimestamp(10));

fl.setArrivalTime(rs.getTimestamp(11));

fl.setAirlineName(rs.getString(12));

System.out.println(tk);

System.out.println(fl);

System.out.println(bk);

BoardingPass myPass = new BoardingPass(tk, bk, fl);

return myPass;

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

private boolean isBoardingPassPrinted(Long ticketId) throws InvalidBoardingPassException{

PreparedStatement ps = null;

ResultSet rs = null;

try {

ps = con.prepareStatement("SELECT state FROM ticket WHERE id = ?");

ps.setLong(1, ticketId);

rs = ps.executeQuery();

while(rs.next()){

if (rs.getInt("state") == 2) {

return true;

} else {

return false;

}

}

throw new InvalidBoardingPassException("Couldn't find a pass by that ticket ID, try again.");

} catch(Exception e) {

System.out.println("Exception caught: " + e);

throw new InvalidBoardingPassException("There was an error creating this boarding pass. Call tech support 1-800-POD-1010");

}

}

private void printBoardingPass(Long ticketID) throws InvalidBoardingPassException {

PreparedStatement ps = null;

int result;

try {

ps = con.prepareStatement("UPDATE ticket SET state = 2, date\_modified = now() WHERE id = ?");

ps.setLong(1, ticketID);

result = ps.executeUpdate();

if (result != 1){

throw new InvalidBoardingPassException("There was an error creating this boarding pass. Call tech support 1-800-POD-1011");

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

throw new InvalidBoardingPassException("There was an error creating this boarding pass. Call tech support 1-800-POD-1012");

}

}

public Ticket modifyTicket(Long ticketId, String seatPreference, String mealPreference, String passengerName) throws InvalidTicketModificationException {

try {

PreparedStatement ps = null;

int rs;

ps = con.prepareStatement("UPDATE ticket SET seat\_pref = ?, meal\_pref = ?, passenger\_name = ? WHERE id = ?");

ps.setString(1, seatPreference);

ps.setString(2, mealPreference);

ps.setString(3, passengerName);

ps.setLong(4, ticketId);

rs = ps.executeUpdate();

if (rs != 1) {

throw new InvalidTicketModificationException("There was an error updating this ticket. Call tech support 1-800-POD-1011");

}

else {

ps = con.prepareStatement("SELECT \* FROM ticket WHERE id = ?");

ps.setLong(1, ticketId);

ResultSet rs2 = ps.executeQuery();

while (rs2.next()) {

Ticket t = new Ticket();

t.setId(rs2.getLong("id"));

t.setFlightId(rs2.getLong("flight\_id"));

t.setBookingId(rs2.getLong("booking\_id"));

t.setSeatType(rs2.getString("seat\_type"));

t.setSeatPref(rs2.getString("seat\_pref"));

t.setMealPref(rs2.getString("meal\_pref"));

t.setPassengerName(rs2.getString("passenger\_name"));

t.setCost(rs2.getLong("cost"));

t.setState(rs2.getLong("state"));

return t;

}

}

return null;

} catch(SQLException e) {

System.out.println("Exception caught: " + e);

throw new InvalidTicketModificationException("There was an error updating this ticket. Call tech support 1-800-POD-1011");

}

}

private void updatePoints(Long bookingId){

PreparedStatement ps = null;

try {

// By using LIMIT 1, we only add one points set into the database even if there are 2 billing records

ps = con.prepareStatement("INSERT INTO points (date\_created, date\_modified, user\_id, billing\_id, amount) " +

"SELECT DISTINCT now(), now(), bill.user\_id, bill.id, bill.amount FROM billing bill WHERE bill.booking\_id = ? AND bill.amount > 0 LIMIT 1");

ps.setLong(1, bookingId);

int val = ps.executeUpdate();

if (val != 1) {

// To few or too many records

throw new InvalidBoardingPassException("No billing information is available for this boarding pass. Contact an agent at 1-800-POD-1013");

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

}

}

ReservationServiceTest.java

package ubc.ece419.pod10;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.\*;

import ubc.ece419.pod10.domain.Billing;

import ubc.ece419.pod10.domain.Flight;

import ubc.ece419.pod10.service.FlightService;

import ubc.ece419.pod10.service.ReservationService;

import ubc.ece419.pod10.service.UserService;

import junit.framework.TestCase;

import com.opensymphony.xwork2.ActionSupport;

//Please run UserServiceTest at least once

public class ReservationServiceTest extends TestCase {

public static Long testUserID;

public static Long testBookingID;

public static Long testFlightID;

public static List<Flight> testFlightList = new ArrayList<Flight>();

public static List<Flight> listOfFlights;

//Boolean continueTest = false;

private static boolean prep = false;

private static boolean testAddBookings = false;

private static boolean testAddTickets = false;

String name = "tester";

String password = "test1234";

String emailAddress = "test@pod10.com";

String creditCardNum = "4500123456785555";

String seatType = "";

String seatPref = "";

String mealPref = "";

UserService userService = new UserService();

FlightService flightService = new FlightService();

ReservationService reservationService = new ReservationService();

public void testPrep() throws Exception

{

Flight fl = new Flight();

prep = true;

try {

testUserID = userService.loginCheck(emailAddress, password);

} catch (Exception e) {

//continueTest = false;

System.out.println(e.getMessage());

fail("==TEST ABORTED: Prep==");

return;

}

if(testUserID == null)

{

//continueTest = false;

System.out.println("==Cannot find User ID with given email and password==");

fail("==Reservation Service TEST ABORTED==");

}

try {

listOfFlights = flightService.getFlightSelection();

} catch (Exception e) {

//continueTest = false;

System.out.println(e.getMessage());

fail("==TEST ABORTED: Get Flight Lists==");

return;

}

if(listOfFlights == null)

{

fail("Null Flight Lists returned");

}

fl = listOfFlights.get(0);

testFlightID = fl.getId();

}

public void testAddBookings() throws Exception {

if(!prep)

{

testPrep();

}

testAddBookings = true;

try {

testBookingID = reservationService.addBooking(testUserID);

} catch (Exception e) {

System.out.println(e.getMessage());

fail("==TEST ABORTED: Test Add Bookings==");

return;

}

if(testBookingID == null)

{

System.out.println("==Cannot find Booking ID with given LoginID==");

fail("==TEST ABORTED: Test Add Bookings==");

}

}

public void testAddTickets() throws Exception

{

if(!prep)

testPrep();

else if(!testAddBookings)

testAddBookings();

testAddTickets= true;

try {

testFlightList = reservationService.addTickets(testUserID, testFlightID, testBookingID, name, seatType, seatPref, mealPref);

} catch (Exception e) {

System.out.println(e.getMessage());

fail("==TEST ABORTED: Prep==");

return;

}

if(testFlightList == null)

{

System.out.println("==Something wrong when adding tickets==");

fail("==TEST ABORTED: Test Add Tickets==");

}

}

public void testBilling() throws Exception

{

if(!prep)

testPrep();

else if(!testAddBookings)

testAddBookings();

if(!testAddTickets)

testAddTickets();

Billing bill\_init = new Billing();

Billing bill = new Billing();

//int ret;

int amount = 2500;

try {

bill\_init = reservationService.addBilling(testUserID, testBookingID, amount, name, creditCardNum, false);

} catch (Exception e) {

System.out.println(e.getMessage());

fail("==TEST ABORTED: Add Billing==");

return;

}

//assertNotNull("Add billing returns 0 or negative", bill\_init);

try {

bill = reservationService.getCurrentBilling(testBookingID);

} catch (Exception e) {

System.out.println(e.getMessage());

fail("==TEST ABORTED: Get Billing==");

return;

}

assertEquals("Billing: Amount not equal", amount, bill.getAmount());

//assertEquals("Billing: UserID not equal", testUserID, bill.getUserId());

assertEquals("Billing: Name not equal", name, bill.getName());

assertEquals("Billing: Credit card # not equal", creditCardNum, bill.getCreditCardNum());

assertFalse("Billing: Loyalty point was not used!", bill.isLoyaltyPointsUsed());

}

}

searchFlightsTest.java

package ubc.ece419.pod10;

import ubc.ece419.pod10.service.FlightService;

import junit.framework.TestCase;

import com.opensymphony.xwork2.ActionSupport;

public class searchFlightsTest extends TestCase {

public void testsomething() throws Exception

{

}

}

SelectManifestAction.java

package ubc.ece419.pod10.action.staff;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Flight;

import ubc.ece419.pod10.domain.Staff;

import ubc.ece419.pod10.domain.User;

import ubc.ece419.pod10.domain.User;

import ubc.ece419.pod10.service.FlightService;

import ubc.ece419.pod10.service.UserService;

import com.opensymphony.xwork2.ActionSupport;

public class SelectManifestAction extends BaseAction{

List<Flight> selectionList = new ArrayList<Flight>();

String userId, staffPlace, message = "";

Long userType;

FlightService flightService = new FlightService();

UserService userService = new UserService();

public String getMessage() {

return message;

}

public void setMessage(String message) {

this.message = message;

}

public Long getUserType() {

return userType;

}

public void setUserType(Long userType) {

this.userType = userType;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public String execute() {

if (userId == null)

return ERROR;

Long userId\_l = Long.parseLong(userId);

//If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || !authenticate(userId\_l)) {

return LOGIN;

}

//check whether the user is a customer, staff, or manager

User user = userService.getUserById(Long.parseLong(userId));

if(user == null) return ERROR;

List<Flight> flights = flightService.getFlightSelection();

userType = user.getUserType();

selectionList.clear();

//decide what to show the user based on what user\_type is

if(userType == 0){

message = "Customers: you may not view any flights.";

//don't allow customers to see the flight manifests

return ERROR;

}

else if(userType == 1){ //user is a staff

//only show flights that have the staff's place as the origin/destination

String flightOrigin, flightDestination;

Staff staff = userService.getStaffByUserId(user.getId());

staffPlace = staff.getPlace();

for(int i = 0; i < flights.size(); i++){

flightDestination = flights.get(i).getDestination();

flightOrigin = flights.get(i).getOrigin();

//message += staffPlace + " " + flightDestination + " " + flightOrigin + "<br>";

message = "Staff: You may view flights where the origin or destination is your location.";

if( flightOrigin.equals(staffPlace) || flightDestination.equals(staffPlace)){

selectionList.add(flights.get(i));

}

}

}

else if(userType == 2){ //user is a manager

message = "Managers: you may view all flights.";

selectionList = flights; //managers see all flights

}

if (selectionList == null) {

return ERROR;

}

return SUCCESS;

}

public List<Flight> getSelectionList() {

return selectionList;

}

public void setSelectionList(List<Flight> selectionList) {

this.selectionList = selectionList;

}

public FlightService getFlightService() {

return flightService;

}

public void setFlightService(FlightService flightService) {

this.flightService = flightService;

}

}

Staff.java

package ubc.ece419.pod10.domain;

public class Staff {

String place;

Long userId, id, placeId;

public Long getUserId() {

return userId;

}

public void setUserId(Long userId) {

this.userId = userId;

}

public Long getId() {

return id;

}

public void setId(Long staffId) {

this.id = staffId;

}

public Long getPlaceId() {

return placeId;

}

public void setPlaceId(Long placeId) {

this.placeId = placeId;

}

public String getPlace() {

return place;

}

public void setPlace(String place) {

this.place = place;

}

}

StaffCancelBookingAction.java

package ubc.ece419.pod10.action.staff;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Billing;

import ubc.ece419.pod10.domain.Ticket;

import ubc.ece419.pod10.service.ReservationService;

import com.opensymphony.xwork2.ActionSupport;

@SuppressWarnings("serial")

public class StaffCancelBookingAction extends BaseAction{

String bookingId;

int cancelState;

String userId;

String customerId;

Billing bill = new Billing();

List<Ticket> ticketList = new ArrayList<Ticket>();

ReservationService reservationService = new ReservationService();

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

Long customerIdLong = Long.parseLong(customerId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (bookingId == null) {

return ERROR;

}

Long bookingIdLong = Long.parseLong(bookingId);

cancelState = reservationService.getBookingCancelState(bookingIdLong);

if (cancelState != 0){

bill = reservationService.getCurrentBilling(bookingIdLong);

ticketList = reservationService.getCurrentTickets(bookingIdLong);

}

return SUCCESS;

}

public String getBookingId() {

return bookingId;

}

public void setBookingId(String bookingId) {

this.bookingId = bookingId;

}

public Billing getBill() {

return bill;

}

public void setBill(Billing bill) {

this.bill = bill;

}

public List<Ticket> getTicketList() {

return ticketList;

}

public void setTicketList(List<Ticket> ticketList) {

this.ticketList = ticketList;

}

public ReservationService getReservationService() {

return reservationService;

}

public void setReservationService(ReservationService reservationService) {

this.reservationService = reservationService;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public int getCancelState() {

return cancelState;

}

public void setCancelState(int cancelState) {

this.cancelState = cancelState;

}

public String getCustomerId() {

return customerId;

}

public void setCustomerId(String customerId) {

this.customerId = customerId;

}

}

StaffConfirmCancelAction.java

package ubc.ece419.pod10.action.staff;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Billing;

import ubc.ece419.pod10.domain.Ticket;

import ubc.ece419.pod10.service.ReservationService;

import com.opensymphony.xwork2.ActionSupport;

public class StaffConfirmCancelAction extends BaseAction {

String bookingId, billingId, userId;

int cancelBooking, cancelTickets, refundBill;

String customerId;

ReservationService reservationService = new ReservationService();

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (bookingId == null) {

return ERROR;

}

Long bookingIdLong = Long.parseLong(bookingId);

Long billingIdLong = Long.parseLong(billingId);

Long customerIdLong = Long.parseLong(customerId);

cancelBooking = reservationService.cancelBooking(bookingIdLong);

cancelTickets = reservationService.cancelTickets(bookingIdLong);

refundBill = reservationService.refundBill(billingIdLong);

return SUCCESS;

}

public String getBookingId() {

return bookingId;

}

public void setBookingId(String bookingId) {

this.bookingId = bookingId;

}

public String getBillingId() {

return billingId;

}

public void setBillingId(String billingId) {

this.billingId = billingId;

}

public int getCancelBooking() {

return cancelBooking;

}

public void setCancelBooking(int cancelBooking) {

this.cancelBooking = cancelBooking;

}

public int getCancelTickets() {

return cancelTickets;

}

public void setCancelTickets(int cancelTickets) {

this.cancelTickets = cancelTickets;

}

public int getRefundBill() {

return refundBill;

}

public void setRefundBill(int refundBill) {

this.refundBill = refundBill;

}

public ReservationService getReservationService() {

return reservationService;

}

public void setReservationService(ReservationService reservationService) {

this.reservationService = reservationService;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public String getCustomerId() {

return customerId;

}

public void setCustomerId(String customerId) {

this.customerId = customerId;

}

}

StaffLoginAction.java

package ubc.ece419.pod10.action.staff;

import ubc.ece419.pod10.action.user.LoginAction;

import ubc.ece419.pod10.service.UserService;

public class StaffLoginAction extends LoginAction {

UserService userService = new UserService();

String password, emailAddress, errorMessage;

Boolean loggedIn = false;

String userId;

public String execute() {

// If both emailAddress and password are null, user just landed on page

if (password == null && emailAddress == null) {

return INPUT;

}

if (password == null || emailAddress == null) {

errorMessage = "Please fill out all the fields";

System.out.println("Please fill out all the fields");

return ERROR;

}

Long userIdFromDb = userService.loginCheck(emailAddress, password);

if (userIdFromDb == null) {

errorMessage = "Incorrect password for input email.";

System.out.println("Incorrect password for input email.");

return ERROR;

}

userId = userIdFromDb.toString();

user = userService.getUserById(Long.parseLong(userId));

if (user.getUserType() == 0) {

return ERROR;

}

// Generate session token and store it in the user table.

try {

userService.updateSessionToken(userIdFromDb);

} catch(Exception e) {

System.out.println("Could not update session");

}

// Redirect to search page.

loggedIn = true;

return SUCCESS;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getEmailAddress() {

return emailAddress;

}

public void setEmailAddress(String emailAddress) {

this.emailAddress = emailAddress;

}

public String getErrorMessage() {

return errorMessage;

}

public void setErrorMessage(String errorMessage) {

this.errorMessage = errorMessage;

}

public Boolean getLoggedIn() {

return loggedIn;

}

public void setLoggedIn(Boolean loggedIn) {

this.loggedIn = loggedIn;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

}

StaffModifyTicketAction.java

package ubc.ece419.pod10.action.staff;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Ticket;

import ubc.ece419.pod10.exceptions.InvalidTicketModificationException;

import ubc.ece419.pod10.service.ReservationService;

import com.opensymphony.xwork2.ActionSupport;

public class StaffModifyTicketAction extends BaseAction {

String ticketId, seatPreference, mealPreference, passengerName, userId;

String customerId;

Ticket ticket;

public Ticket getTicket() {

return ticket;

}

public void setTicket(Ticket ticket) {

this.ticket = ticket;

}

ReservationService reservationService = new ReservationService();

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if(ticketId == null || seatPreference == null || mealPreference == null || passengerName == null) {

return ERROR;

}

Long ticketIdInt = Long.parseLong(ticketId);

Long customerIdLong = Long.parseLong(customerId);

try {

ticket = reservationService.modifyTicket(ticketIdInt, seatPreference, mealPreference, passengerName);

} catch (InvalidTicketModificationException e) {

// TODO Auto-generated catch block

e.printStackTrace();

}

return SUCCESS;

}

public String getModifyTicketGuiInput(){

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (ticketId == null){

return ERROR;

}

return SUCCESS;

}

public String getSeatPreference() {

return seatPreference;

}

public void setSeatPreference(String seatPreference) {

this.seatPreference = seatPreference;

}

public String getMealPreference() {

return mealPreference;

}

public void setMealPreference(String mealPreference) {

this.mealPreference = mealPreference;

}

public String getPassengerName() {

return passengerName;

}

public void setPassengerName(String passengerName) {

this.passengerName = passengerName;

}

public String getticketId() {

return ticketId;

}

public void setticketId(String ticketId) {

this.ticketId = ticketId;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public String getTicketId() {

return ticketId;

}

public void setTicketId(String ticketId) {

this.ticketId = ticketId;

}

public String getCustomerId() {

return customerId;

}

public void setCustomerId(String customerId) {

this.customerId = customerId;

}

public ReservationService getReservationService() {

return reservationService;

}

public void setReservationService(ReservationService reservationService) {

this.reservationService = reservationService;

}

}

StaffViewBookingAction.java

package ubc.ece419.pod10.action.staff;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Booking;

import ubc.ece419.pod10.domain.TicketSet;

import ubc.ece419.pod10.service.ReservationService;

@SuppressWarnings("serial")

public class StaffViewBookingAction extends BaseAction{

String userId, bookingId;

String customerId;

Booking booking = new Booking();

List<TicketSet> ticketSetList = new ArrayList<TicketSet>();

ReservationService reservationService = new ReservationService();

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (userId == null) {

return ERROR;

}

Long bookingIdLong = Long.parseLong(bookingId);

Long customerIdLong = Long.parseLong(customerId);

booking = reservationService.getCurrentBooking(bookingIdLong);

ticketSetList = reservationService.getTicketSetList(bookingIdLong);

return SUCCESS;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public String getBookingId() {

return bookingId;

}

public void setBookingId(String bookingId) {

this.bookingId = bookingId;

}

public Booking getBooking() {

return booking;

}

public void setBooking(Booking booking) {

this.booking = booking;

}

public List<TicketSet> getTicketSetList() {

return ticketSetList;

}

public void setTicketSetList(List<TicketSet> ticketSetList) {

this.ticketSetList = ticketSetList;

}

public ReservationService getReservationService() {

return reservationService;

}

public void setReservationService(ReservationService reservationService) {

this.reservationService = reservationService;

}

public String getCustomerId() {

return customerId;

}

public void setCustomerId(String customerId) {

this.customerId = customerId;

}

}

StaffViewBookingListAction.java

package ubc.ece419.pod10.action.staff;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Booking;

import ubc.ece419.pod10.service.ReservationService;

@SuppressWarnings("serial")

public class StaffViewBookingListAction extends BaseAction{

String userId;

String customerId;

List<Booking> bookingList = new ArrayList<Booking>();

ReservationService reservationService = new ReservationService();

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (userId == null) {

return ERROR;

}

Long userIdLong = Long.parseLong(userId);

Long customerIdLong = Long.parseLong(customerId);

bookingList = reservationService.getBookingList(customerIdLong);

return SUCCESS;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public List<Booking> getBookingList() {

return bookingList;

}

public void setBookingList(List<Booking> bookingList) {

this.bookingList = bookingList;

}

public ReservationService getReservationService() {

return reservationService;

}

public void setReservationService(ReservationService reservationService) {

this.reservationService = reservationService;

}

public String getCustomerId() {

return customerId;

}

public void setCustomerId(String customerId) {

this.customerId = customerId;

}

}

Statistics.java

package ubc.ece419.pod10.domain;

import java.util.Map;

public class Statistics {

public Statistics() {

super();

}

public Statistics(Integer flightsDelayed, Integer flightsEarly, Integer flightsOnTime, Integer flightsFull, Integer flightsHalfEmpty,

Map<String, Integer> flightOriginMap, Map<String, Integer> flightDestinationMap, String mostPopular) {

super();

this.flightsDelayed = flightsDelayed;

this.flightsEarly = flightsEarly;

this.flightsOnTime = flightsOnTime;

this.flightsFull = flightsFull;

this.flightsHalfEmpty = flightsHalfEmpty;

this.flightOriginMap = flightOriginMap;

this.flightDestinationMap = flightDestinationMap;

this.mostPopular = mostPopular;

}

private Integer flightsDelayed;

private Integer flightsEarly;

private Integer flightsOnTime;

private Integer flightsFull;

private Integer flightsHalfEmpty;

private Map<String, Integer> flightOriginMap;

private Map<String, Integer> flightDestinationMap;

private String mostPopular;

public Integer getFlightsDelayed() {

return flightsDelayed;

}

public void setFlightsDelayed(Integer flightsDelayed) {

this.flightsDelayed = flightsDelayed;

}

public Integer getFlightsEarly() {

return flightsEarly;

}

public void setFlightsEarly(Integer flightsEarly) {

this.flightsEarly = flightsEarly;

}

public Integer getFlightsOnTime() {

return flightsOnTime;

}

public void setFlightsOnTime(Integer flightsOnTime) {

this.flightsOnTime = flightsOnTime;

}

public Integer getFlightsFull() {

return flightsFull;

}

public void setFlightsFull(Integer flightsFull) {

this.flightsFull = flightsFull;

}

public Integer getFlightsHalfEmpty() {

return flightsHalfEmpty;

}

public void setFlightsHalfEmpty(Integer flightsHalfEmpty) {

this.flightsHalfEmpty = flightsHalfEmpty;

}

public String getMostPopular() {

return mostPopular;

}

public void setMostPopular(String mostPopular) {

this.mostPopular = mostPopular;

}

public Map<String, Integer> getFlightOriginMap() {

return flightOriginMap;

}

public void setFlightOriginMap(Map<String, Integer> flightOriginMap) {

this.flightOriginMap = flightOriginMap;

}

public Map<String, Integer> getFlightDestinationMap() {

return flightDestinationMap;

}

public void setFlightDestinationMap(Map<String, Integer> flightDestinationMap) {

this.flightDestinationMap = flightDestinationMap;

}

}

Ticket.java

package ubc.ece419.pod10.domain;

import java.sql.Timestamp;

public class Ticket {

public Ticket() {

super();

}

public Ticket(Long id, Timestamp dateCreated, Timestamp dateModified,

Long flightId, Long bookingId, String seatType, String seatPref,

String mealPref, String passengerName, Long cost, Long state) {

super();

this.id = id;

this.dateCreated = dateCreated;

this.dateModified = dateModified;

this.flightId = flightId;

this.bookingId = bookingId;

this.seatType = seatType;

this.seatPref = seatPref;

this.mealPref = mealPref;

this.passengerName = passengerName;

this.cost = cost;

this.state = state;

}

private Long id;

private Timestamp dateCreated;

private Timestamp dateModified;

private Long flightId;

private Long bookingId;

private String seatType;

private String seatPref;

private String mealPref;

private String passengerName;

private Long cost;

private Long state;

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public Timestamp getDateCreated() {

return dateCreated;

}

public void setDateCreated(Timestamp dateCreated) {

this.dateCreated = dateCreated;

}

public Timestamp getDateModified() {

return dateModified;

}

public void setDateModified(Timestamp dateModified) {

this.dateModified = dateModified;

}

public Long getFlightId() {

return flightId;

}

public void setFlightId(Long flightId) {

this.flightId = flightId;

}

public Long getBookingId() {

return bookingId;

}

public void setBookingId(Long bookingId) {

this.bookingId = bookingId;

}

public String getSeatType() {

return seatType;

}

public void setSeatType(String seatType) {

this.seatType = seatType;

}

public String getSeatPref() {

return seatPref;

}

public void setSeatPref(String seatPref) {

this.seatPref = seatPref;

}

public String getMealPref() {

return mealPref;

}

public void setMealPref(String mealPref) {

this.mealPref = mealPref;

}

public String getPassengerName() {

return passengerName;

}

public void setPassengerName(String passengerName) {

this.passengerName = passengerName;

}

public Long getCost() {

return cost;

}

public void setCost(Long cost) {

this.cost = cost;

}

public Long getState() {

return state;

}

public void setState(Long state) {

this.state = state;

}

}

TicketSet.java

package ubc.ece419.pod10.domain;

public class TicketSet {

public TicketSet() {

super();

}

public TicketSet(Ticket ticket, Flight flight) {

super();

this.ticket = ticket;

this.flight = flight;

}

private Ticket ticket;

private Flight flight;

public Ticket getTicket() {

return ticket;

}

public void setTicket(Ticket ticket) {

this.ticket = ticket;

}

public Flight getFlight() {

return flight;

}

public void setFlight(Flight flight) {

this.flight = flight;

}

}

Transactions.java

package ubc.ece419.pod10.domain;

import java.sql.Timestamp;

public class Transactions {

public Transactions() {

super();

}

public Transactions(Long billingId, Long amount, String name, String creditCardNumber, Timestamp dateCreated, String reason, Long pointsAwarded) {

super();

this.billingId = billingId;

this.amount = amount;

this.name = name;

this.creditCardNumber = creditCardNumber;

this.dateCreated = dateCreated;

this.reason = reason;

this.pointsAwarded = pointsAwarded;

}

private Long billingId;

private Long amount;

private String name;

private String creditCardNumber;

private Timestamp dateCreated;

private String reason;

private Long pointsAwarded;

public Long getBillingId() {

return billingId;

}

public void setBillingId(Long billingId) {

this.billingId = billingId;

}

public Long getAmount() {

return amount;

}

public void setAmount(Long amount) {

this.amount = amount;

}

public String getCreditCardNumber() {

return creditCardNumber;

}

public void setCreditCardNumber(String creditCardNumber) {

this.creditCardNumber = creditCardNumber;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Timestamp getDateCreated() {

return dateCreated;

}

public void setDateCreated(Timestamp dateCreated) {

this.dateCreated = dateCreated;

}

public String getReason() {

return reason;

}

public void setReason(String reason) {

this.reason = reason;

}

public Long getPointsAwarded() {

return pointsAwarded;

}

public void setPointsAwarded(Long pointsAwarded) {

this.pointsAwarded = pointsAwarded;

}

}

UnableToCreateUserException.java

package ubc.ece419.pod10.exceptions;

public class UnableToCreateUserException extends Exception {

public UnableToCreateUserException() {

super();

}

public UnableToCreateUserException(String s) {

super(s);

}

public UnableToCreateUserException(Throwable t, String s) {

super(s, t);

}

public UnableToCreateUserException(Throwable t) {

super(t);

}

}

User.java

package ubc.ece419.pod10.domain;

import java.sql.Timestamp;

public class User {

private String emailAddress, password, sessionToken, name, salt;

private Long id, currentPointsBalance;

private Long userType;

private Timestamp dateCreated, dateModified, sessionExpirationDate;

public Long getUserType() {

return userType;

}

public void setUserType(Long l) {

this.userType = l;

if(this.userType == null) throw new NullPointerException();

}

public Timestamp getSessionExpirationDate() {

return sessionExpirationDate;

}

public void setSessionExpirationDate(Timestamp sessionExpirationDate) {

this.sessionExpirationDate = sessionExpirationDate;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getEmailAddress() {

return emailAddress;

}

public void setEmailAddress(String emailAddress) {

this.emailAddress = emailAddress;

}

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public Timestamp getDateCreated() {

return dateCreated;

}

public void setDateCreated(Timestamp dateCreated) {

this.dateCreated = dateCreated;

}

public Timestamp getDateModified() {

return dateModified;

}

public void setDateModified(Timestamp dateModified) {

this.dateModified = dateModified;

}

public String getSessionToken() {

return sessionToken;

}

public void setSessionToken(String sessionToken) {

this.sessionToken = sessionToken;

}

public Long getCurrentPointsBalance() {

return currentPointsBalance;

}

public void setCurrentPointsBalance(Long currentPointsBalance) {

this.currentPointsBalance = currentPointsBalance;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getSalt() {

return salt;

}

public void setSalt(String salt) {

this.salt = salt;

}

}

UserReserveFlightsAction.java

package ubc.ece419.pod10.action.user;

import com.opensymphony.xwork2.ActionSupport;

import java.sql.Timestamp;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Billing;

import ubc.ece419.pod10.domain.Flight;

import ubc.ece419.pod10.domain.Ticket;

import ubc.ece419.pod10.domain.User;

import ubc.ece419.pod10.service.FlightService;

import ubc.ece419.pod10.service.ReservationService;

public class UserReserveFlightsAction extends BaseAction{

//query output

String passengerName;

List<Flight> flightList;

List<Ticket> ticketList;

String seatType;

//query input

String flightId;

String userId;

String bookingId;

String mealPreference;

String seatPreference;

String origin;

String destination;

Billing billing;

String creditCardName;

String creditCardNum;

String isPayByPoints;

String canPayByPoints;

String subtotal;

String availPoints;

ReservationService reservationService = new ReservationService();

/\*\*

\* Use Case: Reserve Flights

\* query database for the reserved Flight information, returned data will be displayed on the website

\* @return a boolean value indicates query database is successful or not

\*/

public String execute(){

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (flightId == null || passengerName == null || seatType == null || seatPreference == null || mealPreference == null) {

return ERROR;

}

Long flightId\_l = Long.parseLong(flightId);

Long bookingId\_l = reservationService.addBooking(userId\_l);

bookingId = bookingId\_l.toString();

flightList = reservationService.addTickets(userId\_l, flightId\_l, bookingId\_l, passengerName, seatType, seatPreference, mealPreference);

String origin\_code = flightList.get(0).getOrigin();

origin = reservationService.getPlaceByCode(origin\_code);

String destination\_code = flightList.get(0).getDestination();

destination = reservationService.getPlaceByCode(destination\_code);

return SUCCESS;

}

/\*\*

\* Use Case: Reserve Flights

\* Add tickets entry to the database and query database for the reserved Flight information, , returned data will be displayed on the website

\* @return a boolean value indicates query database is successful or not

\*/

public String executeLoop(){

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (flightId == null || bookingId == null || passengerName == null || seatType == null || seatPreference == null || mealPreference == null) {

return ERROR;

}

Long flightId\_l = Long.parseLong(flightId);

Long bookingId\_l = Long.parseLong(bookingId);

flightList = reservationService.addTickets(userId\_l, flightId\_l, bookingId\_l, passengerName, seatType, seatPreference, mealPreference);

String origin\_code = flightList.get(0).getOrigin();

origin = reservationService.getPlaceByCode(origin\_code);

String destination\_code = flightList.get(0).getDestination();

destination = reservationService.getPlaceByCode(destination\_code);

return SUCCESS;

}

/\*\*

\* Use Case: Manage Payment

\* add an entry to Billing Table, and query the bill information for web display

\* @return a boolean value indicates query database is successful or not

\*/

public String makePayment() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (bookingId == null || subtotal == null || isPayByPoints == null)

return ERROR;

Long bookingId\_l = Long.parseLong(bookingId);

Integer subtotal\_i = Integer.parseInt(subtotal);

int isPayByPoints\_i = Integer.parseInt(isPayByPoints);

boolean isPayByPoints\_b = false;

if (isPayByPoints\_i == 1)

isPayByPoints\_b = true;

if (isPayByPoints\_b) {

creditCardName = null;

creditCardNum = null;

}

else

{

if (creditCardName == null || creditCardNum == null ) {

return ERROR;

}

}

billing = reservationService.addBilling(userId\_l, bookingId\_l, subtotal\_i, creditCardName, creditCardNum, isPayByPoints\_b);

return SUCCESS;

}

/\*\*

\* Use Case: Manage Payments

\* Get payment input and check if royalpoint can be used to make payment

\* @return a boolean value indicates query database is successful or not

\*/

public String getBillingGuiInput() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (bookingId == null)

return ERROR;

Long bookingId\_l = Long.parseLong(bookingId);

boolean canPayByPoints\_b = reservationService.canUsePoints(userId\_l, bookingId\_l);

canPayByPoints = Boolean.toString(canPayByPoints\_b);

availPoints = Integer.toString(reservationService.getPointsBalance(userId\_l));

int subtotal\_i = reservationService.getBookingSubtotal(bookingId\_l);

subtotal = Integer.toString(subtotal\_i);

ticketList = reservationService.getCurrentTickets(bookingId\_l);

return SUCCESS;

}

/\*\*

\* Use Case: Reserve Flights

\* check if userId and flightId are valid to reserve a ticket

\* @return a boolean value indicates query database is successful or not

\*/

public String getReserveGuiInput(){

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (flightId == null ){

return ERROR;

}

return SUCCESS;

}

/\*\*

\* Use Case: Reserve Flights

\* check if userId and flightId are valid to reserve a ticket

\* @return a boolean value indicates query database is successful or not

\*/

public String getReserveGuiInputLoop() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (flightId == null || bookingId == null){

return ERROR;

}

return SUCCESS;

}

public String getMealPreference() {

return mealPreference;

}

public void setMealPreference(String mealPreference) {

this.mealPreference = mealPreference;

}

public String getSeatPreference() {

return seatPreference;

}

public void setSeatPreference(String seatPreference) {

this.seatPreference = seatPreference;

}

public String getPassengerName() {

return passengerName;

}

public void setPassengerName(String passengerName) {

this.passengerName = passengerName;

}

public List<Flight> getFlightList() {

return flightList;

}

public void setFlightList(List<Flight> flightList) {

this.flightList = flightList;

}

public String getSeatType() {

return seatType;

}

public void setSeatType(String seatType) {

this.seatType = seatType;

}

public String getFlightId() {

return flightId;

}

public void setFlightId(String flightId) {

this.flightId = flightId;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public String getBookingId() {

return bookingId;

}

public void setBookingId(String bookingId) {

this.bookingId = bookingId;

}

public Billing getBilling() {

return billing;

}

public void setBilling(Billing billing) {

this.billing = billing;

}

public String getCreditCardName() {

return creditCardName;

}

public void setCreditCardName(String creditCardName) {

this.creditCardName = creditCardName;

}

public String getCreditCardNum() {

return creditCardNum;

}

public void setCreditCardNum(String creditCardNum) {

this.creditCardNum = creditCardNum;

}

public String getIsPayByPoints() {

return isPayByPoints;

}

public void setIsPayByPoints(String isPayByPoints) {

this.isPayByPoints = isPayByPoints;

}

public String getCanPayByPoints() {

return canPayByPoints;

}

public void setCanPayByPoints(String canPayByPoints) {

this.canPayByPoints = canPayByPoints;

}

public List<Ticket> getTicketList() {

return ticketList;

}

public void setTicketList(List<Ticket> ticketList) {

this.ticketList = ticketList;

}

public String getSubtotal() {

return subtotal;

}

public void setSubtotal(String subtotal) {

this.subtotal = subtotal;

}

public String getOrigin() {

return origin;

}

public void setOrigin(String origin) {

this.origin = origin;

}

public String getDestination() {

return destination;

}

public void setDestination(String destination) {

this.destination = destination;

}

public String getAvailPoints() {

return availPoints;

}

public void setAvailPoints(String availPoints) {

this.availPoints = availPoints;

}

}

UserSearchFlightsAction.java

package ubc.ece419.pod10.action.user;

import java.sql.Timestamp;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Flight;

import ubc.ece419.pod10.domain.User;

import ubc.ece419.pod10.service.FlightService;

import com.opensymphony.xwork2.ActionSupport;

public class UserSearchFlightsAction extends BaseAction {

String origin, destination, requiredSeats, departureTime, seatType, bookingId, userId, returnTime;

List<Flight> flightList;

List<Flight> flightList\_Return;

FlightService flightService = new FlightService();

/\*\*

\* Use Case: Search Flights

\* Use userInput to query database for Flights, a list of Flight object is returned for web display

\* @return a boolean value indicates query database is successful or not

\*/

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (origin == null || destination == null || requiredSeats == null || departureTime == null || seatType == null

|| origin.isEmpty() || destination.isEmpty() || requiredSeats.isEmpty() || departureTime.isEmpty() || seatType.isEmpty()) {

return ERROR;

}

Timestamp departureTimestamp = Timestamp.valueOf(departureTime+" 00:00:00");

Integer requiredSeatsInt = Integer.parseInt(requiredSeats);

boolean isFirstClass\_b = false;

if (seatType.equals("1"))

isFirstClass\_b = true;

flightList = flightService.getFlights(origin, destination, departureTimestamp, requiredSeatsInt, isFirstClass\_b);

// if returnTime is null, return flights are not set

if (returnTime != null && !returnTime.isEmpty()){

Timestamp returnTimestamp = Timestamp.valueOf(returnTime+" 00:00:00");

flightList\_Return = flightService.getFlights(destination, origin, returnTimestamp, requiredSeatsInt, isFirstClass\_b);

}

return SUCCESS;

}

/\*\*

\* Use Case: Search Flights

\* Use userInput to query database for Flights, a list of Flight object is returned for web display

\* @return a boolean value indicates query database is successful or not

\*/

public String executeLoop() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (origin == null || destination == null || bookingId == null || requiredSeats == null || departureTime == null || seatType == null

|| bookingId.isEmpty() || origin.isEmpty() || destination.isEmpty() || requiredSeats.isEmpty() || departureTime.isEmpty() || seatType.isEmpty()) {

return ERROR;

}

Timestamp departureTimestamp = Timestamp.valueOf(departureTime+" 00:00:00");

Integer requiredSeatsInt = Integer.parseInt(requiredSeats);

boolean isFirstClass\_b = false;

if (seatType.equals("1"))

isFirstClass\_b = true;

flightList = flightService.getFlights(origin, destination, departureTimestamp, requiredSeatsInt, isFirstClass\_b);

// if returnTime is null, return flights are not set

if (returnTime != null && !returnTime.isEmpty()){

Timestamp returnTimestamp = Timestamp.valueOf(returnTime+" 00:00:00");

flightList\_Return = flightService.getFlights(destination, origin, returnTimestamp, requiredSeatsInt, isFirstClass\_b);

}

return SUCCESS;

}

/\*\*

\* Use Case: Search Flights

\* check if userId is valid

\* @return a boolean value indicates valid or not

\*/

public String getSearchGuiInput() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

return SUCCESS;

}

/\*\*

\* Use Case: Search Flights

\* check if userId and bookingId are valid

\* @return a boolean value indicates userId and bookingId validation

\*/

public String getSearchGuiInputLoop() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (bookingId == null){

return ERROR;

}

return SUCCESS;

}

public String getSearchReturnGuiInput() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (bookingId == null || origin == null || destination == null || origin.isEmpty() || destination.isEmpty()){

return ERROR;

}

return SUCCESS;

}

public String getOrigin() {

return origin;

}

public void setOrigin(String origin) {

this.origin = origin;

}

public String getDestination() {

return destination;

}

public void setDestination(String destination) {

this.destination = destination;

}

public List<Flight> getFlightList() {

return flightList;

}

public void setFlightList(List<Flight> flightList) {

this.flightList = flightList;

}

public String getRequiredSeats() {

return requiredSeats;

}

public void setRequiredSeats(String requiredSeats) {

this.requiredSeats = requiredSeats;

}

public String getDepartureTime() {

return departureTime;

}

public void setDepartureTime(String departureTime) {

this.departureTime = departureTime;

}

public String getSeatType() {

return seatType;

}

public void setSeatType(String seatType) {

this.seatType = seatType;

}

public String getBookingId() {

return bookingId;

}

public void setBookingId(String bookingId) {

this.bookingId = bookingId;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public String getReturnTime() {

return returnTime;

}

public void setReturnTime(String returnTime) {

this.returnTime = returnTime;

}

public List<Flight> getFlightList\_Return() {

return flightList\_Return;

}

public void setFlightList\_Return(List<Flight> flightList\_Return) {

this.flightList\_Return = flightList\_Return;

}

}

UserService.java

package ubc.ece419.pod10.service;

import java.math.BigInteger;

import java.security.MessageDigest;

import java.security.NoSuchAlgorithmException;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.sql.Timestamp;

import java.util.ArrayList;

import java.util.List;

import java.util.UUID;

import ubc.ece419.pod10.domain.Staff;

import ubc.ece419.pod10.domain.Transactions;

import ubc.ece419.pod10.domain.User;

import ubc.ece419.pod10.exceptions.UnableToCreateUserException;

import com.opensymphony.xwork2.ActionSupport;

public class UserService extends ActionSupport{

private static final String URL = "jdbc:mysql://localhost:3306/pod10\_db?zeroDateTimeBehavior=convertToNull";

private static final String DRIVER\_NAME = "com.mysql.jdbc.Driver";

private static final String DB\_USER\_NAME = "root";

private static final String DB\_PASSWORD = "ECE419\_pod10";

Connection con = null;

Statement stmt = null;

MessageDigest messageDigest = null;

public UserService() {

super();

try {

// Set up database connection

Class.forName(DRIVER\_NAME).newInstance();

con = DriverManager.getConnection(URL, DB\_USER\_NAME, DB\_PASSWORD);

// Set up MessageDigest

messageDigest = MessageDigest.getInstance("MD5");

} catch(InstantiationException e) {

e.printStackTrace();

} catch(IllegalAccessException e) {

e.printStackTrace();

} catch(ClassNotFoundException e) {

e.printStackTrace();

} catch (SQLException e) {

e.printStackTrace();

} catch (NoSuchAlgorithmException e) {

e.printStackTrace();

}

}

/\*\*

\* Checks whether email address is already in database or not.

\* @param emailAddress The email address to check for.

\* @return False if email address is not already used, true otherwise.

\*/

public Boolean isEmailAddressAlreadyUsed(String emailAddress) {

try {

ResultSet rs = null;

PreparedStatement ps = con.prepareStatement("SELECT \* FROM User WHERE email\_address = ?");

ps.setString(1, emailAddress);

rs = ps.executeQuery();

if (rs == null) {

// getting a null back is illegal, so returning true.

return true;

} else {

if (rs.next()) {

return true;

} else {

return false;

}

}

} catch (SQLException e) {

e.printStackTrace();

}

return true;

}

/\*\*

\* Creates a user with given name, password and email address.

\* @param name The name of the user.

\* @param password The password that the user selects.

\* @param emailAddress The email address of the user.

\* @throws UnableToCreateUserException Thrown when user cannot be created. This could be because of a non-unique

\* email address.

\*/

public Long createUser(String name, String password, String emailAddress) throws UnableToCreateUserException {

try {

if (isEmailAddressAlreadyUsed(emailAddress)) {

throw new UnableToCreateUserException("Email address is already used. Please try some other email address.");

}

String salt = UUID.randomUUID().toString().substring(0,16);

String saltedPassword = salt + password;

messageDigest.update(saltedPassword.getBytes(), 0, saltedPassword.length());

String password\_hashed = (new BigInteger(1, messageDigest.digest())).toString(16).substring(0,16);

String sessionToken = generateRandomSessionToken();

Timestamp tomorrow = new Timestamp(System.currentTimeMillis() + 86400000);

PreparedStatement ps = con.prepareStatement("INSERT INTO User (name, password, salt, email\_address, session\_token, session\_expiry\_date) VALUES (?, ?, ?, ?, ?, ?)");

ps.setString(1, name);

ps.setString(2, password\_hashed);

ps.setString(3, salt);

ps.setString(4, emailAddress);

ps.setString(5, sessionToken);

ps.setTimestamp(6, tomorrow);

int result = ps.executeUpdate();

if (result != 1) {

throw new UnableToCreateUserException("Unable to create user for " + name);

}

// Get userId and return it.

PreparedStatement ps2 = con.prepareStatement("SELECT \* from User WHERE email\_address=?");

ps2.setString(1, emailAddress);

ResultSet rs = ps2.executeQuery();

rs.next();

return rs.getLong("id");

} catch(SQLException e) {

System.out.println("Exception caught: " + e);

}

return null;

}

/\*\*

\* Logs in a user and returns user id if successful.

\* @param emailAddress Email Address of user.

\* @param password Password that user entered.

\* @return userId if the matching email address and password are entered. Null otherwise.

\*/

public Long loginCheck(String emailAddress, String password) {

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM User WHERE email\_address = ?");

ps.setString(1, emailAddress);

ResultSet rs = ps.executeQuery();

rs.next();

String db\_password = rs.getString("password");

String db\_salt = rs.getString("salt");

String saltedPassword = db\_salt + password;

messageDigest.update(saltedPassword.getBytes(), 0, saltedPassword.length());

String passwordHashed = (new BigInteger(1, messageDigest.digest())).toString(16).substring(0,16);

if (passwordHashed.equals(db\_password)) {

return rs.getLong("id");

}

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

/\*\*

\*

\* @param userId

\* @return List of transactions returned from database. Null if no valid transactions are found.

\*/

public List<Transactions> getTransactions(Long userId) {

List<Transactions> transactionList = new ArrayList<Transactions>();

try{

PreparedStatement ps, ps1 = null;

ResultSet rs, rs1 = null;

ps = con.prepareStatement("SELECT \* FROM billing WHERE user\_id = ?");

ps.setLong(1, userId);

rs = ps.executeQuery();

while (rs.next()) {

Transactions trans = new Transactions();

long billingId = rs.getLong("id");

trans.setBillingId(billingId);

trans.setDateCreated(rs.getTimestamp("date\_created"));

if(!rs.getBoolean("is\_loyalty\_points\_used")) {

trans.setAmount(rs.getLong("amount"));

String fullCard = rs.getString("credit\_card\_number");

trans.setCreditCardNumber("XXXX XXXX XXXX " + fullCard.substring(fullCard.length() - 4));

trans.setName(rs.getString("name"));

} else{

trans.setCreditCardNumber(null);

trans.setName(null);

}

ps1 = con.prepareStatement("SELECT \* FROM points WHERE billing\_id = ?");

ps1.setLong(1, billingId);

rs1 = ps1.executeQuery();

if (rs.getBoolean("is\_loyalty\_points\_used")) {

// If Loyalty points have been used, query the points table for the amount of points used, and the reason.

rs1.next();

trans.setAmount(rs1.getLong("amount"));

trans.setReason(rs1.getString("reason"));

} else {

// If loyalty points have not been used, check the points table for any entries for that billing. If there are any entries,

// then get amount and reason from that entry, else display message saying that you will be awarded when you print the boarding pass.

if (rs1.next()) {

trans.setPointsAwarded(rs1.getLong("amount"));

trans.setReason(rs1.getString("reason"));

} else {

trans.setPointsAwarded(null);

if (trans.getAmount() < 0)

trans.setReason("Booking Cancelled");

else

trans.setReason("Points will be awarded after boarding pass is printed.");

}

}

transactionList.add(trans);

}

return transactionList;

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return null;

}

public int getPoints(Long userId) {

int points;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM user WHERE id = ?");

ps.setLong(1, userId);

ResultSet rs = ps.executeQuery();

rs.next();

points = rs.getInt("current\_point\_balance");

System.out.println(points);

return points;

} catch(Exception e) {

System.out.println("Exception caught: " + e);

}

return -1;

}

/\*\*

\* Checks if a user's session has expired or not.

\* @param userId Identifies the user.

\* @return True if session has expired, false otherwise

\*/

public Boolean hasValidSession(Long userId) {

User user = getUserById(userId);

Long now = System.currentTimeMillis();

if (user.getSessionToken() != null && user.getSessionExpirationDate().getTime() > (now)) {

return true;

}

return false;

}

/\*\*

\* Updates the session token for user.

\* @param userId Identifies the user.

\* @throws Exception When no user can be found for userId

\*/

public void updateSessionToken(Long userId) throws Exception {

String sessionToken = generateRandomSessionToken();

Timestamp tomorrow = new Timestamp(System.currentTimeMillis() + 86400000);

try {

PreparedStatement ps = con.prepareStatement("UPDATE User SET session\_token=?, session\_expiry\_date=? WHERE id=?");

ps.setString(1, sessionToken);

ps.setTimestamp(2, tomorrow);

ps.setLong(3, userId);

int result = ps.executeUpdate();

if (result != 1) {

throw new Exception("Could not find user " + userId);

}

} catch(SQLException e) {

e.printStackTrace();

}

}

public User getUserById(Long userId) {

User user = null;

try {

PreparedStatement ps = con.prepareStatement("SELECT \* FROM user where id = ?");

ps.setLong(1, userId);

ResultSet rs = ps.executeQuery();

rs.next();

user = new User();

user.setId(rs.getLong("id"));

user.setEmailAddress(rs.getString("email\_address"));

user.setPassword(rs.getString("password"));

user.setName(rs.getString("name"));

user.setSessionToken(rs.getString("session\_token"));

user.setSessionExpirationDate(rs.getTimestamp("session\_expiry\_date"));

user.setCurrentPointsBalance(rs.getLong("current\_point\_balance"));

user.setUserType(rs.getLong("user\_type"));

user.setDateCreated(rs.getTimestamp("date\_created"));

user.setDateModified(rs.getTimestamp("date\_modified"));

} catch(SQLException e) {

e.printStackTrace();

}

return user;

}

//gets the entry from the staff table associated with a user

//an entry only exists if the user\_type is 1 or 2 (staff or management)

public Staff getStaffByUserId(long userId)

{

Staff staff = new Staff();

try{

//query the staff table

PreparedStatement ps = con.prepareStatement("SELECT \* FROM staff where user\_id = ?");

ps.setLong(1, userId);

ResultSet rs = ps.executeQuery();

if(!rs.next())

return null;

staff.setId(rs.getLong("id"));

staff.setPlaceId(rs.getLong("place\_id"));

staff.setUserId(userId);

//query the place table for the place name

ps = con.prepareStatement("SELECT \* FROM place where id = ?");

ps.setLong(1, staff.getPlaceId());

rs = ps.executeQuery();

if(!rs.next())

return null;

staff.setPlace(rs.getString("code"));

} catch(SQLException e) {

e.printStackTrace();

}

return staff;

}

public String generateRandomSessionToken() {

Timestamp now = new Timestamp(System.currentTimeMillis());

String sessionTokenString = now.toString();

messageDigest.update(sessionTokenString.getBytes(), 0, sessionTokenString.length());

return (new BigInteger(1, messageDigest.digest())).toString(16).substring(0,16);

}

}

UserServiceTest.java

package ubc.ece419.pod10;

import junit.framework.TestCase;

import ubc.ece419.pod10.action.user.UserSearchFlightsAction;

import ubc.ece419.pod10.domain.User;

import ubc.ece419.pod10.exceptions.UnableToCreateUserException;

import ubc.ece419.pod10.service.UserService;

import com.opensymphony.xwork2.ActionSupport;

public class UserServiceTest extends TestCase {

public static Long testUserID = (long) 0;

public static Boolean continueTest = false;

String name = "tester";

String password = "test1234";

String wrongpass = "hahaha";

String emailAddress = "test@pod10.com";

UserService userService = new UserService();

public void testCreateUser() throws Exception

{

boolean result;

boolean user\_not\_exist = true;

try {

testUserID = userService.createUser(name, password, emailAddress);

}

catch(UnableToCreateUserException e){

System.out.println(e.getMessage());

user\_not\_exist = false;

}

catch (Exception e) {

continueTest = false;

System.out.println(e.getMessage());

fail("==TEST ABORTED: Create User ID==");

return;

}

if(user\_not\_exist){

assertTrue("Create User: Cannot create test User", (testUserID > 0));

}

else

{

testUserID = userService.loginCheck(emailAddress, password);

}

continueTest = true; //can continue to next test from this point

// email address should be "blocked" since already used

result = userService.isEmailAddressAlreadyUsed(emailAddress);

assertTrue("Create User: Email address incorrectly shown as unused", result);

}

public void testLoginUser() throws Exception

{

Long userID;

// Check inputting wrong password

userID = userService.loginCheck(emailAddress, wrongpass);

assertNull("Login User: Wrong password detected as correct", userID);

// Check inputting correct password obtains correct id

userID = userService.loginCheck(emailAddress, password);

assertEquals("Login User: Wrong user ID returned with correct password", testUserID, userID);

}

public void testUserByID() throws Exception

{

User retUser;

testUserID = userService.loginCheck(emailAddress, password);

retUser = userService.getUserById(testUserID);

//Check for parameters

assertEquals(name, retUser.getName());

assertEquals(emailAddress, retUser.getEmailAddress());

//assertEquals(password, retUser.getPassword());

}

}

ViewBoardingPassAction.java

package ubc.ece419.pod10.action.user;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.BoardingPass;

import ubc.ece419.pod10.exceptions.InvalidBoardingPassException;

import ubc.ece419.pod10.exceptions.InvalidBookingException;

import ubc.ece419.pod10.exceptions.InvalidTicketException;

import ubc.ece419.pod10.exceptions.ViewedTooEarlyException;

import ubc.ece419.pod10.service.ReservationService;

@SuppressWarnings("serial")

public class ViewBoardingPassAction extends BaseAction {

String userId;

String bookingId;

String message;

List<BoardingPass> boardingPassList = new ArrayList<BoardingPass>();

ReservationService reservationService = new ReservationService();

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || !authenticate(Long.parseLong(userId))) {

return LOGIN;

}

if (bookingId == null) {

return ERROR;

}

Long bookingIdLong = Long.parseLong(bookingId);

try {

boardingPassList = reservationService.getBoardingPass(bookingIdLong);

} catch (InvalidTicketException e){

setMessage("Invalid boarding ticket ID");

} catch (ViewedTooEarlyException e){

setMessage("This boarding pass cannot be printed yet");

} catch (InvalidBoardingPassException e) {

e.printStackTrace();

} catch (InvalidBookingException e) {

e.printStackTrace();

}

return SUCCESS;

}

// Automatically generated getters and setters

public List<BoardingPass> getBoardingPassList() {

return boardingPassList;

}

public void setBoardingPassList(List<BoardingPass> boardingPassList) {

this.boardingPassList = boardingPassList;

}

public String getMessage() {

return message;

}

public void setMessage(String message) {

this.message = message;

}

public String getBookingId() {

return bookingId;

}

public void setBookingId(String bookingId) {

this.bookingId = bookingId;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

}

ViewBookingAction.java

package ubc.ece419.pod10.action.user;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Booking;

import ubc.ece419.pod10.domain.TicketSet;

import ubc.ece419.pod10.service.ReservationService;

@SuppressWarnings("serial")

public class ViewBookingAction extends BaseAction{

String userId, bookingId;

Booking booking = new Booking();

List<TicketSet> ticketSetList = new ArrayList<TicketSet>();

ReservationService reservationService = new ReservationService();

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (userId == null) {

return ERROR;

}

Long bookingIdLong = Long.parseLong(bookingId);

booking = reservationService.getCurrentBooking(bookingIdLong);

ticketSetList = reservationService.getTicketSetList(bookingIdLong);

return SUCCESS;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public String getBookingId() {

return bookingId;

}

public void setBookingId(String bookingId) {

this.bookingId = bookingId;

}

public Booking getBooking() {

return booking;

}

public void setBooking(Booking booking) {

this.booking = booking;

}

public List<TicketSet> getTicketSetList() {

return ticketSetList;

}

public void setTicketSetList(List<TicketSet> ticketSetList) {

this.ticketSetList = ticketSetList;

}

public ReservationService getReservationService() {

return reservationService;

}

public void setReservationService(ReservationService reservationService) {

this.reservationService = reservationService;

}

}

ViewBookingListAction.java

package ubc.ece419.pod10.action.user;

import java.util.ArrayList;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Booking;

import ubc.ece419.pod10.service.ReservationService;

@SuppressWarnings("serial")

public class ViewBookingListAction extends BaseAction{

String userId;

List<Booking> bookingList = new ArrayList<Booking>();

ReservationService reservationService = new ReservationService();

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

if (userId == null) {

return ERROR;

}

Long userIdLong = Long.parseLong(userId);

bookingList = reservationService.getBookingList(userIdLong);

return SUCCESS;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public List<Booking> getBookingList() {

return bookingList;

}

public void setBookingList(List<Booking> bookingList) {

this.bookingList = bookingList;

}

public ReservationService getReservationService() {

return reservationService;

}

public void setReservationService(ReservationService reservationService) {

this.reservationService = reservationService;

}

}

ViewedTooEarlyException.java

package ubc.ece419.pod10.exceptions;

public class ViewedTooEarlyException extends Exception{

public ViewedTooEarlyException() {

super();

}

public ViewedTooEarlyException(String s) {

super(s);

}

public ViewedTooEarlyException(Throwable t, String s) {

super(s, t);

}

public ViewedTooEarlyException(Throwable t) {

super(t);

}

}

ViewFlightManifestTest.java

package ubc.ece419.pod10;

import ubc.ece419.pod10.action.staff.ViewManifestAction;

import ubc.ece419.pod10.service.FlightService;

import ubc.ece419.pod10.action.user.UserSearchFlightsAction;

import junit.framework.TestCase;

import com.opensymphony.xwork2.ActionSupport;

public class ViewFlightManifestTest extends TestCase {

//TCF.100.05 - Retrieve list of accessible flight manifests

public void testListFlightManifests() throws Exception

{

}

}

ViewManifestAction.java

package ubc.ece419.pod10.action.staff;

import java.util.List;

import ubc.ece419.pod10.domain.Ticket;

import ubc.ece419.pod10.service.FlightService;

import com.opensymphony.xwork2.ActionSupport;

public class ViewManifestAction extends ActionSupport {

String flightManifest\_flightId;

List<Ticket> ticketList;

FlightService flightService = new FlightService();

String userId;

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public String execute() {

if (flightManifest\_flightId == null) {

return ERROR;

}

Long flightManifest\_flightIdInt = Long.parseLong(flightManifest\_flightId);

ticketList = flightService.getFlightManifest(flightManifest\_flightIdInt);

return SUCCESS;

}

public List<Ticket> getTicketList() {

return ticketList;

}

public void setFlightList(List<Ticket> ticketList) {

this.ticketList = ticketList;

}

public String getFlightManifest\_flightId() {

return flightManifest\_flightId;

}

public void setFlightManifest\_flightId(String flightManifest\_flightId) {

this.flightManifest\_flightId = flightManifest\_flightId;

}

}

ViewStatisticsAction.java

package ubc.ece419.pod10.action.staff;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Statistics;

import ubc.ece419.pod10.service.FlightService;

import com.opensymphony.xwork2.ActionSupport;

public class ViewStatisticsAction extends BaseAction {

String userId;

FlightService flightService = new FlightService();

Statistics Stat;

public String execute() {

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || userId.isEmpty()) {

return LOGIN;

}

Long userId\_l = Long.parseLong(userId);

if (!authenticate(userId\_l)) {

return LOGIN;

}

Stat = flightService.getStatistics();

return SUCCESS;

}

public Statistics getStat() {

return Stat;

}

public void setStat(Statistics Stat) {

this.Stat = Stat;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

}

ViewTransactionHistoryAction.java

package ubc.ece419.pod10.action.user;

import java.util.List;

import ubc.ece419.pod10.action.BaseAction;

import ubc.ece419.pod10.domain.Transactions;

import ubc.ece419.pod10.domain.User;

import ubc.ece419.pod10.service.UserService;

import com.opensymphony.xwork2.ActionSupport;

public class ViewTransactionHistoryAction extends BaseAction {

UserService userService = new UserService();

String userId;

List<Transactions> transactionList;

int points;

public String execute() {

Long userId\_l = Long.parseLong(userId);

// If userId is null, or if user cannot be authenticated, re-direct user to the login page.

if (userId == null || !authenticate(userId\_l)) {

return LOGIN;

}

if (userId == null) {

return ERROR;

}

Long userIdInt = Long.parseLong(userId);

transactionList = userService.getTransactions(userIdInt);

points = userService.getPoints(userIdInt);

return SUCCESS;

}

public UserService getUserService() {

return userService;

}

public void setUserService(UserService userService) {

this.userService = userService;

}

public String getUserId() {

return userId;

}

public void setUserId(String userId) {

this.userId = userId;

}

public List<Transactions> getTransactionList() {

return transactionList;

}

public void setTransactionList(List<Transactions> transactionList) {

this.transactionList = transactionList;

}

public int getPoints() {

return points;

}

public void setPoints(int points) {

this.points = points;

}

}

airplan\_css.css

html,body,div,span,applet,object,iframe,h1,h2,h3,h4,h5,h6,p,blockquote,button,pre,a,abbr,acronym,address,big,cite,code,del,dfn,em,font,img,ins,kbd,q,s,samp,small,strike,strong,sub,sup,tt,var,dl,dt,dd,ol,ul,li,fieldset,form,input,textarea,label,legend,table,caption,tbody,tfoot,thead,tr,th,td{margin:0;padding:0;border:0;outline:0;font-weight:inherit;font-style:inherit;font-size:100.01%;font-family:inherit;vertical-align:baseline}

:focus{outline:0}

body{line-height:1}

ol,ul{list-style:none}

div{margin:0 auto}

table{border-collapse:collapse;border-spacing:0;width:100%}

fieldset{border:1px solid #111;padding:3em;margin:0 auto}

caption,th,td{text-align:left;font-weight:normal}

blockquote:before,blockquote:after,q:before,q:after{content:""}

blockquote,q{quotes:"" ""}

.dn{display:none}

ol,ul{list-style-image:none;list-style-position:outside;list-style-type:none}

ol.disc,ul.disc{list-style-image:none;list-style-position:outside;list-style-type:disc;padding-left:15px}

ol.circle,ul.circle{list-style-image:none;list-style-position:outside;list-style-type:circle;padding-left:15px}

ol.decimal,ul.decimal{list-style-image:none;list-style-position:outside;list-style-type:decimal;padding-left:20px}

em,i,.em{font-style:italic}

strong,b,.strong{font-weight:bold}

.u{text-decoration:underline}

.lower{text-transform:lowercase}

.upper{text-transform:uppercase}

.strike{text-decoration:line-through}

.alt-font,.alt-font-lt,.alt-font-med,.alt-font-hvy{font-family:'AauxNextWide-SemiBold','Lucida Grande','Lucida Sans Unicode',sans-serif}

.alt-font-hvy{font-weight:bold}

.alt-font-serif{font-family:Georgia,serif}

.default-font{font-family:'Helvetica','Lucida Grande','Lucida Sans Unicode',sans-serif}

.fs-xxs{font-size:10px}

.fs-xs{font-size:11px}

.fs-n{font-size:12px}

.fs-s{font-size:13px}

.fs-m{font-size:14px}

.fs-l{font-size:16px}

.fs-xl{font-size:20px}

.fs-xxl{font-size:25px}

.fs-xxxl{font-size:35px}

.lh-xs{line-height:12px}

.lh-s{line-height:14px}

.lh-m{line-height:16px}

.lh-l{line-height:18px}

.lh-xl{line-height:20px}

.lh-xxl{line-height:24px}

.lh-xxxl{line-height:28px}

.lh-n{line-height:normal}

.lh-48{line-height:48px}

a,a:link,a:visited,a:hover,a:active{text-decoration:none;cursor:pointer;color:#333}

a:hover{text-decoration:underline}

a.u,a.u:link,a.u:visited,a.u:hover,a.u:active{text-decoration:underline}

a.u:hover{text-decoration:none}

a.nodec:hover,.nodec{text-decoration:none}

.c1,a.c1{color:#111}

a.c1:hover,a.c1:active{color:#aaa}

.c2,a.c2{color:#666}

a.c2:hover,a.c2:active{color:#fff}

.m-basic{margin:18px}

.m-basic-t{margin-top:18px}

.m-basic-r{margin-right:18px}

.m-basic-b{margin-bottom:18px}

.m-basic-l{margin-left:18px}

.m-xs-t{margin-top:1px}

.m-s{margin:2px}

.m-s-t{margin-top:2px}

.m-s-r{margin-right:2px}

.m-s-b{margin-bottom:2px}

.m-s-l{margin-left:2px}

.m-m{margin:4px}

.m-m-t{margin-top:4px}

.m-m-r{margin-right:4px}

.m-m-b{margin-bottom:4px}

.m-m-l{margin-left:4px}

.m-l{margin:8px}

.m-l-t{margin-top:8px}

.m-l-r{margin-right:8px}

.m-l-b{margin-bottom:8px}

.m-l-l{margin-left:8px}

.m-xl{margin:12px}

.m-xl-t{margin-top:12px}

.m-xl-r{margin-right:12px}

.m-xl-b{margin-bottom:12px}

.m-xl-l{margin-left:12px}

.p-basic{padding:18px}

.p-basic-t{padding-top:18px}

.p-basic-r{padding-right:18px}

.p-basic-b{padding-bottom:18px}

.p-basic-l{padding-left:18px}

.p-s{padding:2px}

.p-s-t{padding-top:2px}

.p-s-r{padding-right:2px}

.p-s-b{padding-bottom:2px}

.p-s-l{padding-left:2px}

.p-m{padding:4px}

.p-m-t{padding-top:4px}

.p-m-r{padding-right:4px}

.p-m-b{padding-bottom:4px}

.p-m-l{padding-left:4px}

.p-l{padding:8px}

.p-l-t{padding-top:8px}

.p-l-r{padding-right:8px}

.p-l-b{padding-bottom:8px}

.p-l-l{padding-left:8px}

.p-xl{padding:12px}

.p-xl-t{padding-top:12px}

.p-xl-r{padding-right:12px}

.p-xl-b{padding-bottom:12px}

.p-xl-l{padding-left:12px}

.p-8-l{padding-left:8px}

.p-20-r{padding-right:20px}

.p-20-l{padding-left:20px}

.p-20-t{padding-top:20px}

.p-27-t{padding-top:27px}

.p-27-r{padding-right:27px}

.p-27-l{padding-left:27px}

.p-40{padding:40px}

.p-40-r{padding-right:40px}

.p-40-b{padding-bottom:40px}

.p-40-l{padding-left:40px}

.p-40-t{padding-top:40px}

.p-72-l{padding-left:72px}

.p-72-t{padding-top:72px}

.p-100-l{padding-left:100px}

.p-100-t{padding-top:100px}

.m-6-r{margin-right:6px}

.m-8-b{margin-bottom:8px}

.m-10-t{margin-top:10px}

.m-10-b{margin-bottom:10px}

.m-10-r{margin-right:10px}

.m-16-r{margin-right:16px}

.m-18-l{margin-left:18px}

.m-20-t{margin-top:20px}

.m-20-b{margin-bottom:20px}

.m-20-r{margin-right:20px}

.m-20-l{margin-left:20px}

.m-30-t{margin-top:30px}

.m-30-l{margin-left:30px}

.m-60-l{margin-left:60px}

.m-60-r{margin-right:60px}

.ta-c{text-align:center}

.ta-l{text-align:left}

.ta-r{text-align:right}

.fl-c{float:center}

.fl-l{float:left}

.fl-r{float:right}

.bg-full{width:100%;height:100%;position:absolute;top:0;left:0;z-index:0}

.bg{height:500px;background:#fff}

.bg-header{background-color:#248DBA}

.bg-white{background-color:#fff}

.bg-gray{background-color:#f1f1f0}

.bg-lblue{background-color:#5bf}

textarea.no-resize{resize:none}

input{display:inline-block;padding:1px;margin:1px}

input[type="radio"]{background:url(http://ryanfait.com/resources/custom-checkboxes-and-radio-buttons/radio.png) no-repeat;}

.inputs{float:left;width:200px}

.inputs-wide{width:350px}

label{display:inline-block;padding:.24em 0}

.labels{float:left;width:108px}

.labels-wide{float:left;width:150px}

legend{padding:0 10px}

select{background-color:#fff;border:1px solid #e1e1e1;padding:4px}

html{height:100%;width:100%}

body{height:100%;width:100%;font:12px 'Lucida Grande','Lucida Sans Unicode',sans-serif;color:#5d5d5d;font-style:normal;-webkit-text-size-adjust:none}

.wrap{width:1024px;margin:0 auto;background-color:#fff}

#wrap-border{width:960px;border-left:7px solid #262626;border-right:7px solid #262626;margin:0 auto;background-color:#fff}

.content-1024{position:relative;overflow:hidden;width:1024px}

.col-1024{width:1024px;float:center;position:relative}

.col-768{width:768px;float:center;position:relative}

.col-600{width:600px;float:center;position:relative;padding:25px}

.col-480{width:480px;float:center;position:relative;padding:50px 25px}

.header{height:110px;width:1024px;margin:auto;position:relative;z-index:99;background:#248DBA}

.header-logo{float:left;width:222px;margin-top:20px}

.header-links{float:left;width:762px;margin-right:20px}

.header-pers{width:762px;height:80px;overflow:hidden}

.header-menus{width:762px;height:20px;margin-top:3px;overflow:hidden}

.header-banner{float:left;height:50px;width:400px}

.header-user-links{float:right;height:33px;overflow:hidden}

.header-menu{float:left;text-align:center;width:213px}

.header-user-link{float:left;text-transform:uppercase;margin-left:12px;margin-top:16px}

.menu{overflow:hidden}

.menu a.mm-item-link,.menu a.mm-item-link:link,.menu a.mm-item-link:visited,.menu a.mm-item-link:hover,.menu a.mm-item-link:active{display:inline-block;position:relative;text-decoration:none;z-index:11;color:#fff}

.menu a.mm-item-link-sel,.menu a.mm-item-link-sel:link,.menu a.mm-item-link-sel:visited,.menu a.mm-item-link-sel:hover,.menu a.mm-item-link-sel:active{display:inline-block;position:relative;text-decoration:none;z-index:11;color:#248DBA}

.menu.top-menu a.mm-item-link,.menu.top-menu a.mm-item-link:link,.menu.top-menu a.mm-item-link:visited,.menu.top-menu a.mm-item-link:hover,.menu.top-menu a.mm-item-link:active{z-index:14}

.header-panel{width:245px}

.fixed-thing{position:fixed}

#corner-rnd{border-radius:25px}

.row-focus{font-family:"Helvetica","Lucida Sans Unicode","Lucida Grande",Sans-Serif;width:960px;border-collapse:collapse;margin:20px}

.row-focus th{font-weight:normal;color:#14a;padding:5px 5px}

.row-focus td{color:#76a;border-top:1px solid #eef;padding:9px 12px}

.row-focus tr:hover td{color:#339;background:#eff2ff}

.row-focus-fancy{font-family:"Helvetica","Lucida Sans Unicode","Lucida Grande",Sans-Serif;width:960px;border-collapse:collapse;background:url("two\_glossy\_clouds.png") 600px 120px no-repeat;margin:20px}

.row-focus-fancy th{font-weight:normal;color:#14a;padding:5px 5px}

.row-focus-fancy td{color:#65a;border-top:1px solid #eef;padding:9px 12px}

.row-focus-fancy tbody td{background:url("table\_back.png")}

.row-focus-fancy tbody tr:hover td{color:#339;background:none}

.t\_tbl{width:450px}

.textbox-gen{width:200px;border:1px solid #777}

.btn-gen{border:1px solid #777;background-color:#eee;border-radius:3px;padding:3px 10px;text-shadow:1px 1px #dde}

.btn-gen:hover{background-color:#ddd}

.btn-gen:active{background-color:#eff;top:1px;position:relative}

.ftr{width:1000px;margin:0px auto}

billing.sql

CREATE TABLE billing

(id bigint not null auto\_increment,

date\_created timestamp not null,

date\_modified timestamp not null,

user\_id bigint not null,

booking\_id bigint not null,

amount bigint not null,

name varchar(100) default null,

credit\_card\_number varchar(250) default null,

is\_loyalty\_points\_used tinyint(1) not null default 0,

primary key (id),

FOREIGN KEY (user\_id) REFERENCES user(id),

FOREIGN KEY (booking\_id) REFERENCES booking(id)

);

booking.sql

CREATE TABLE booking(

id bigint not null auto\_increment,

date\_created timestamp not null,

date\_modified timestamp not null,

user\_id bigint not null,

cancel\_state tinyint(1) not null default 1,

FOREIGN KEY (user\_id) REFERENCES user(id),

PRIMARY KEY (id)

);

flight.sql

CREATE TABLE flight

(id bigint not null auto\_increment,

date\_created timestamp not null,

date\_modified timestamp not null,

origin varchar(20) not null,

destination varchar(20) not null,

departure\_time timestamp not null,

arrival\_time timestamp not null,

actual\_departure\_time timestamp null default null,

actual\_arrival\_time timestamp null default null,

airline\_name varchar(20) not null,

total\_first\_class int not null default 50,

first\_class\_avail int not null default 50,

total\_economy int not null default 200,

economy\_avail int not null default 200,

base\_cost int not null,

primary key (id) );

import.sql

source 20121013/init.sql

source 20121013/user.sql

source 20121013/booking.sql

source 20121013/flight.sql

source 20121013/ticket.sql

source 20121013/billing.sql

source 20121013/points.sql

source 20121013/place.sql

source 20121013/staff.sql

source 20121013/test\_data2.sql

source 20121013/test\_data.sql

init.sql

drop database pod10\_db;

create database pod10\_db;

use pod10\_db;

place.sql

CREATE TABLE place

(id bigint not null auto\_increment,

date\_created timestamp not null,

date\_modified timestamp not null,

name varchar(40) not null,

code varchar(10) not null,

is\_first\_class\_lounge\_avail boolean not null,

primary key (id) );

points.sql

CREATE TABLE points

(id bigint not null auto\_increment,

date\_created timestamp not null,

date\_modified timestamp not null,

user\_id bigint not null,

billing\_id bigint not null,

amount int not null,

reason varchar(50) not null,

primary key (id),

FOREIGN KEY (user\_id) REFERENCES user(id),

FOREIGN KEY (billing\_id) REFERENCES billing(id)

);

staff.sql

CREATE TABLE staff

(id bigint not null auto\_increment,

user\_id bigint not null,

place\_id bigint not null,

primary key (id),

FOREIGN KEY (user\_id) REFERENCES user(id),

FOREIGN KEY (place\_id) REFERENCES place(id)

);

test\_data.sql

INSERT INTO user (name, email\_address, password, salt, current\_point\_balance, user\_type) VALUES

("user1", "user1@hotmail.com", "46f06d8e2be990ad", "eaa4fba3-5009-48", 0, 0),

("user2", "user2@hotmail.com", "7a1ecf56da38e807", "a5c2470b-bdb7-49", 1000000, 0),

("admin1", "admin1@ryanair.com", "20ed23c5d8680726", "f64a16e9-7580-4e", 0, 1),

("admin2", "admin2@ryanair.com", "334600456bd2548a", "e2a077d9-b253-41", 0, 1),

("admin3", "admin3@ryanair.com", "58cd2f67c8ea29b6", "5dbfabfc-f5ac-44", 0, 1),

("manager1", "manager1@ryanair.com", "beb8f6e8a249ea47", "09e91f27-8a79-4a", 0, 2),

("manager2", "manager2@ryanair.com", "528e22c11ce490e9", "b2bf399b-bed1-42", 0, 2);

INSERT INTO booking(date\_created, date\_modified, user\_id) VALUES

(now(), now(), 1),

(now(), now(), 2);

INSERT INTO ticket (flight\_id, booking\_id, seat\_type, seat\_pref, meal\_pref, passenger\_name, cost, state) VALUES

(1, 1, 1, 'No Preference', 'No Preference', 'Ryan', 599, 1),

(2, 1, 1, 'No Preference', 'No Preference', 'Anuj', 599, 1),

(3, 1, 1, 'No Preference', 'No Preference', 'Yonatan', 599, 1),

(4, 1, 1, 'No Preference', 'No Preference', 'Scott', 599, 1),

(5, 1, 1, 'No Preference', 'No Preference', 'Nathaniel', 599, 1),

(6, 1, 1, 'No Preference', 'No Preference', 'Danny', 599, 1),

(7, 1, 1, 'No Preference', 'No Preference', 'Kevin', 599, 1),

(8, 1, 1, 'No Preference', 'No Preference', 'Louie', 599, 1),

(9, 1, 1, 'No Preference', 'No Preference', 'Shirley', 599, 1),

(10, 2, 2, 'No Preference', 'No Preference', 'Ryan', 2995, 1),

(11, 2, 2, 'No Preference', 'No Preference', 'Anuj', 2995, 1),

(12, 2, 2, 'No Preference', 'No Preference', 'Yonatan', 2995, 1),

(13, 2, 2, 'No Preference', 'No Preference', 'Scott', 2995, 1),

(14, 2, 2, 'No Preference', 'No Preference', 'Nathaniel', 2995, 1),

(15, 2, 2, 'No Preference', 'No Preference', 'Danny', 2995, 1),

(16, 2, 2, 'No Preference', 'No Preference', 'Kevin', 2995, 1),

(17, 2, 2, 'No Preference', 'No Preference', 'Louie', 2995, 1),

(18, 2, 2, 'No Preference', 'No Preference', 'Shirley', 2995, 1);

INSERT INTO billing (user\_id, booking\_id, amount, name, credit\_card\_number, is\_loyalty\_points\_used) VALUES

(1, 1, 5391, "Anuj", "4111111111111111", 0),

(2, 2, 26955, "Not Anuj", "4111111111111112", 0);

INSERT INTO staff (user\_id, place\_id) VALUES

(3, 28),

(4, 21),

(5, 1);

test\_data2.sql

INSERT INTO place (date\_created,date\_modified,name,code,is\_first\_class\_lounge\_avail) VALUES

(NOW(),NOW(),"Atlanta","ATL", 1),

(NOW(),NOW(),"Chicago","ORD", 1),

(NOW(),NOW(),"London","LHR", 1),

(NOW(),NOW(),"Tokyo","HND", 1),

(NOW(),NOW(),"Los Angeles","LAS", 1),

(NOW(),NOW(),"Frankfurt","FRA", 1),

(NOW(),NOW(),"Paris","CDG", 1),

(NOW(),NOW(),"Phoenix","PHX", 1),

(NOW(),NOW(),"Madrid","MAD", 1),

(NOW(),NOW(),"Houston","IAH", 1),

(NOW(),NOW(),"Hong Kong","HKG", 1),

(NOW(),NOW(),"Bangkok","DMK", 1),

(NOW(),NOW(),"San Francisco","SFO", 1),

(NOW(),NOW(),"New York","JFK", 1),

(NOW(),NOW(),"Singapore","SIN", 1),

(NOW(),NOW(),"Osaka","KIX", 1),

(NOW(),NOW(),"Shanghai","PVG", 1),

(NOW(),NOW(),"Beijing","PEK", 1),

(NOW(),NOW(),"Seattle","SEA", 1),

(NOW(),NOW(),"Orlando","MCO", 1),

(NOW(),NOW(),"Toronto","YYZ", 1),

(NOW(),NOW(),"Calgary","YYC", 1),

(NOW(),NOW(),"Edmonton","YEG", 1),

(NOW(),NOW(),"Kelowna","YLW", 1),

(NOW(),NOW(),"Montreal","YUL", 1),

(NOW(),NOW(),"Ottawa","YOW", 1),

(NOW(),NOW(),"Saskatoon","YXE", 1),

(NOW(),NOW(),"Vancouver","YVR", 1),

(NOW(),NOW(),"Whitehorse","YXY", 1),

(NOW(),NOW(),"Winnipeg","YWG", 1),

(NOW(),NOW(),"Yellowknife","YZF", 1);

INSERT INTO flight (date\_created,date\_modified,origin,destination,departure\_time,arrival\_time,actual\_departure\_time,actual\_arrival\_time,airline\_name,total\_first\_class,first\_class\_avail,total\_economy,economy\_avail,base\_cost) VALUES

(NOW(), NOW(), 'ATL', 'YVR', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ATL', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ATL', 'YVR', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ATL', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ATL', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ATL', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ORD', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ORD', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ORD', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LHR', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LHR', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LHR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LAS', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LAS', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LAS', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YYZ', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YYZ', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YUL', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YUL', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YUL', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YOW', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YOW', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YOW', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YXY', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YXY', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YXY', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SIN', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SIN', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SIN', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PEK', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PEK', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PEK', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'KIX', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'KIX', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YVR', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'KIX', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YVR', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'JFK', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YVR', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'JFK', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'JFK', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YVR', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PVG', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YVR', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PVG', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PVG', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YVR', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'CDG', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YVR', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'CDG', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'CDG', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YVR', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HKG', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YVR', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HKG', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HKG', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YVR', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SEA', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YVR', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SEA', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YVR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SEA', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ATL', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ATL', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ATL', 'YVR', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ATL', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ATL', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ATL', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ORD', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ORD', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'ORD', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LHR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LHR', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LHR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LAS', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LAS', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'LAS', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YYZ', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YUL', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YUL', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YUL', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YOW', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YOW', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YOW', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YXY', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YXY', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'YXY', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SIN', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SIN', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SIN', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PEK', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PEK', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PEK', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'KIX', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'KIX', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YVR', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'KIX', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YVR', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'JFK', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YVR', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'JFK', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'JFK', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PVG', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YVR', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PVG', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'PVG', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HND', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'CDG', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YVR', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'CDG', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'CDG', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HKG', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YVR', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HKG', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'HKG', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YVR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SEA', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YVR', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SEA', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YVR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YVR', 'SEA', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ATL', 'YYZ', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ATL', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ATL', 'YYZ', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ATL', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ATL', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ATL', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ORD', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ORD', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YYZ', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ORD', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LHR', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LHR', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YYZ', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LHR', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LAS', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LAS', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YYZ', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LAS', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YUL', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YUL', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YYZ', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YUL', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YOW', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YOW', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YYZ', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YOW', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YXY', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YXY', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YYZ', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YXY', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SIN', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SIN', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YYZ', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SIN', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PEK', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PEK', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YYZ', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PEK', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'KIX', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'KIX', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YYZ', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'KIX', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YYZ', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'JFK', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YYZ', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'JFK', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'JFK', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YYZ', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PVG', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YYZ', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PVG', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PVG', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YYZ', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'CDG', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YYZ', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'CDG', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'CDG', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YYZ', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HKG', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YYZ', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HKG', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HKG', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YYZ', NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 1 DAY, NOW()+INTERVAL 1 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SEA', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YYZ', NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 2 DAY, NOW()+INTERVAL 2 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SEA', NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 4 DAY, NOW()+INTERVAL 4 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YYZ', NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 3 DAY, NOW()+INTERVAL 3 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SEA', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ATL', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ATL', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ATL', 'YYZ', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ATL', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ATL', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ATL', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ORD', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ORD', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YYZ', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'ORD', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'ORD', 'YYZ', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LHR', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LHR', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YYZ', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LHR', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LHR', 'YYZ', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LAS', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LAS', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YYZ', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'LAS', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'LAS', 'YYZ', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YUL', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YUL', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YYZ', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YUL', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YUL', 'YYZ', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YOW', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YOW', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YYZ', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YOW', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YOW', 'YYZ', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YXY', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YXY', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YYZ', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'YXY', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YXY', 'YYZ', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SIN', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SIN', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YYZ', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SIN', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SIN', 'YYZ', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PEK', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PEK', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YYZ', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PEK', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PEK', 'YYZ', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'KIX', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'KIX', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YYZ', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'KIX', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'KIX', 'YYZ', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'JFK', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YYZ', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'JFK', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'JFK', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'JFK', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PVG', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YYZ', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PVG', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'PVG', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'PVG', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HND', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HND', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'CDG', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YYZ', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'CDG', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'CDG', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'CDG', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HKG', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YYZ', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HKG', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'HKG', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'HKG', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YYZ', NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 5 DAY, NOW()+INTERVAL 5 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SEA', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YYZ', NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 6 DAY, NOW()+INTERVAL 6 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SEA', NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 8 DAY, NOW()+INTERVAL 8 DAY+INTERVAL 1 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'SEA', 'YYZ', NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 7 DAY, NOW()+INTERVAL 7 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599),

(NOW(), NOW(), 'YYZ', 'SEA', NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 1 HOUR, NOW()+INTERVAL 9 DAY, NOW()+INTERVAL 9 DAY+INTERVAL 2 HOUR, 'Ryan Air', default, default, default, default, 599);

ticket.sql

-- For state:

-- 0 = Reserved

-- 1 = Paid

-- 2 = Boarding Pass Printed

-- 3 = Cancelled

CREATE TABLE ticket(

id bigint not null auto\_increment,

flight\_id bigint not null,

booking\_id bigint not null,

date\_created timestamp not null default now(),

date\_modified timestamp not null,

seat\_type varchar(30) not null,

seat\_pref varchar(30) not null,

meal\_pref varchar(30) not null,

passenger\_name varchar(100) not null,

cost bigint not null,

state int not null,

FOREIGN KEY (flight\_id) REFERENCES flight(id),

FOREIGN KEY (booking\_id) REFERENCES booking(id),

PRIMARY KEY(id)

);

user.sql

CREATE TABLE user(

id bigint not null auto\_increment,

date\_created timestamp not null default now(),

date\_modified timestamp not null,

name varchar(100) not null,

email\_address varchar(100) not null,

salt varchar(100) not null,

password varchar(1000) not null,

current\_point\_balance int not null default 0,

user\_type int not null default 0,

session\_token varchar(16) null default null,

session\_expiry\_date timestamp null default null,

PRIMARY KEY(id)

);

404NotFoundError.ftl

<html>

<head>

<title>Welcome to Airplan</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/tbg.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="searchFlightsGui.action?userId=${userId}">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-768">

<img src="imgs/emptyroom404.jpg" />

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

BACKUP2viewBookings.ftl

<html>

<head>

<title>View Bookings</title>

<link rel="stylesheet" type="text/css" href="css/airplan\_css.css" />

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font">

<div class="header-logo">

<a href="test\_template.html" title="AirPlan">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</a>

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

<img src="css/pictoptest.png"/>

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

<a class="mm-item-link c9" href="#">Sign In</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="test\_template.html">Home</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="">Reservations</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="">Loyalty Points</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="">Manage</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<table id="row-focus" class="ta-c fs-n">

<thead>

<tr>

<th>Booking ID</th>

<th>Date Created</th>

</tr>

</thead>

<#if bookingSetList??>

<#list bookingSetList as b>

<tbody>

<#if b.booking?? && b.booking.id?? && b.booking.dateCreated??>

<tr>

<td>${b.booking.id}</td>

<td>${b.booking.dateCreated}</td>

</tr>

</#if>

</tbody>

</#list>

<#else>

Bookings Were Not Displayed

</#if>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-723">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

BACKUPviewBookings.ftl

<html>

<head>

<title>View Bookings</title>

<link rel="stylesheet" type="text/css" href="css/airplan\_css.css" />

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font">

<div class="header-logo">

<a href="test\_template.html" title="AirPlan">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</a>

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

<img src="css/pictoptest.png"/>

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

<a class="mm-item-link c9" href="#">Sign In</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="test\_template.html">Home</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="">Reservations</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="">Loyalty Points</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="">Manage</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<table id="row-focus" class="ta-c fs-n">

<thead>

<tr>

<th>Passenger Name</th>

<th>Ticket ID</th>

<th>Flight ID</th>

<th>Booking ID</th>

<th>Seat Type</th>

<th>Seat Preference</th>

<th>Meal Preference</th>

<th>Flight Origin</th>

<th>Flight Destination</th>

<th>Departure Time</th>

<th>Arrival Time Time</th>

<th>Airline Name</th>

</tr>

</thead>

<#if boardingPass??>

<tbody>

<#if boardingPass.myTicket?? && boardingPass.myTicket.id?? && boardingPass.myTicket.seatType?? && boardingPass.myTicket.seatPref?? && boardingPass.myTicket.mealPref?? && boardingPass.myTicket.passengerName?? && boardingPass.myFlight.id?? && boardingPass.myBooking.id?? && boardingPass.myFlight.origin?? && boardingPass.myFlight.destination?? && boardingPass.myFlight.departureTime?? && boardingPass.myFlight.arrivalTime?? && boardingPass.myFlight.airlineName??>

<p>This is a paragraph</p>

<tr>

<td>${boardingPass.myTicket.passengerName}</td>

<td>${boardingPass.myTicket.id}</td>

<td>${boardingPass.myFlight.id}</td>

<td>${boardingPass.myBooking.id}</td>

<td>${boardingPass.myTicket.seatType}</td>

<td>${boardingPass.myTicket.seatPref}</td>

<td>${boardingPass.myTicket.mealPref}</td>

<td>${boardingPass.myFlight.origin}</td>

<td>${boardingPass.myFlight.destination}</td>

<td>${boardingPass.myFlight.departureTime}</td>

<td>${boardingPass.myFlight.arrivalTime}</td>

<td>${boardingPass.myFlight.airlineName}</td>

</tr>

</#if>

<#else>

Bookings Were Not Displayed

</#if>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-723">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

boardingPass.ftl

<html>

<head>

<title>Print Boarding Pass</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/2.png)">

<div class="header-logo">

<a href="test\_template.html" title="AirPlan">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</a>

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="searchFlightsGui.action?userId=${userId}">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<table id="pass\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Passenger Name</th>

<th>Ticket ID</th>

<th>Flight ID</th>

<th>Booking ID</th>

<th>Seat Type</th>

<th>Seat Preference</th>

<th>Meal Preference</th>

<th>Flight Origin</th>

<th>Flight Destination</th>

<th>Departure Time</th>

<th>Arrival Time</th>

<th>Airline Name</th>

</tr>

</thead>

<#list boardingPassList as boardingPass>

<tbody>

<#if boardingPass.myTicket?? && boardingPass.myTicket.id?? && boardingPass.myTicket.seatType?? && boardingPass.myTicket.seatPref?? && boardingPass.myTicket.mealPref?? && boardingPass.myTicket.passengerName?? && boardingPass.myFlight.id?? && boardingPass.myBooking.id?? && boardingPass.myFlight.origin?? && boardingPass.myFlight.destination?? && boardingPass.myFlight.departureTime?? && boardingPass.myFlight.arrivalTime?? && boardingPass.myFlight.airlineName??>

<tr>

<td>${boardingPass.myTicket.passengerName}</td>

<td>${boardingPass.myTicket.id}</td>

<td>${boardingPass.myFlight.id}</td>

<td>${boardingPass.myBooking.id}</td>

<td>${boardingPass.myTicket.seatType}</td>

<td>${boardingPass.myTicket.seatPref}</td>

<td>${boardingPass.myTicket.mealPref}</td>

<td>${boardingPass.myFlight.origin}</td>

<td>${boardingPass.myFlight.destination}</td>

<td>${boardingPass.myFlight.departureTime}</td>

<td>${boardingPass.myFlight.arrivalTime}</td>

<td>${boardingPass.myFlight.airlineName}</td>

</tr>

</#if>

</#list>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

cancelBooking.ftl

<html>

<head>

<title>Cancel Booking</title>

<link rel="stylesheet" type="text/css" href="css/airplan\_css.css" />

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/2.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="searchFlightsGui.action?userId=${userId}">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<#if cancelState??>

<#if cancelState==0>

<div id="corner-rnd" class="col-600 bg-gray">The ticket cannot be canceled because boarding pass is already printed!!!</div>

<#else>

<div id="corner-rnd" class="col-600 bg-gray">Billing summary of selected booking:</div>

<table id="b\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Bill ID</th>

<th>Date Created</th>

<th>Amount</th>

<th>Name</th>

<th>Credit Card</th>

<th>Points Used?</th>

</tr>

</thead>

<#if bill??>

<tbody>

<#if bill.id?? && bill.dateCreated?? && bill.amount?? && bill.isLoyaltyPointsUsed??>

<tr>

<td>${bill.id}</td>

<td>${bill.dateCreated}</td>

<td>${bill.amount}</td>

<td><#if bill.name??>${bill.name}<#else>-</#if></td>

<td><#if bill.creditCardNum??>${bill.creditCardNum}<#else>-</#if></td>

<td>${bill.isLoyaltyPointsUsed()?string("yes", "no")}</td>

</tr>

</#if>

</tbody>

<#else>

Bill was not set

</#if>

</table>

<div id="corner-rnd" class="col-600 bg-gray">This booking contains the following reservations:</div>

<table id="r\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Ticket ID</th>

<th>Flight ID</th>

<th>Seat Type</th>

<th>Seat Preference</th>

<th>Meal Preference</th>

<th>Passenger Name</th>

</tr>

</thead>

<#if ticketList??>

<#list ticketList as ticket>

<tbody>

<#if ticket.id?? && ticket.flightId?? && ticket.seatType?? && ticket.seatPref?? && ticket.mealPref?? && ticket.passengerName??>

<tr>

<td>${ticket.id}</td>

<td>${ticket.flightId}</td>

<td>${ticket.seatType}</td>

<td>${ticket.seatPref}</td>

<td>${ticket.mealPref}</td>

<td>${ticket.passengerName}</td>

</tr>

</#if>

</tbody>

</#list>

<#else>

Tickets were not set

</#if>

</table>

<div class="p-basic col-768">

<a href="confirmCancel.action?userId=${userId}&bookingId=${bookingId}&billingId=${bill.id}"><button type="submit" class="positive btn-gen fl-r">Confirm Cancellation</button></a>

</div>

</#if>

</#if>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

confirmCancel.ftl

<html>

<head>

<title>Confirm Booking Cancellation</title>

<link rel="stylesheet" type="text/css" href="css/airplan\_css.css" />

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/2.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="searchFlightsGui.action?userId=${userId}">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

The booking has been cancelled.

<#if cancelBooking == 1 && cancelTickets != 0 && refundBill == 1>

<p>Booking ${bookingId} has been cancelled.</p>

<p>Bill ${billingId} has been refunded.</p>

<p>${cancelTickets} tickets have been cancelled.</p>

<#else>

There was an error canceling this booking.

</#if>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

flightManifest.ftl

<html>

<head>

<title>View Selected Flight Manifest</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/2.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*USERNAME\* &nbsp; &nbsp; <a class="mm-item-link c9" href="staffSignIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="" onclick="return notDone()">Manage User Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="selectManifest.action?userId=${userId}">View Flight Manifests</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewStatistics.action?userId=${userId}">View Flight Statistics</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<table id="vm\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>ID</th>

<th>Flight ID</th>

<th>Booking ID</th>

<th>Seat Type</th>

<th>Seat Preference</th>

<th>Meal Preference</th>

<th>Passenger Name</th>

</tr>

</thead>

<#if ticketList??>

<#list ticketList as ticket>

<tbody>

<#if ticket.id?? && ticket.flightId?? && ticket.bookingId?? && ticket.seatType?? && ticket.seatPref?? && ticket.mealPref?? && ticket.passengerName??>

<tr>

<td>${ticket.id}</td>

<td>${ticket.flightId}</td>

<td>${ticket.bookingId}</td>

<td>${ticket.seatType}</td>

<td>${ticket.seatPref}</td>

<td>${ticket.mealPref}</td>

<td>${ticket.passengerName}</td>

</tr>

</#if>

</tbody>

</#list>

<#else>

Flight Manifest was not set

</#if>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

InternalServerError.ftl

<html>

<head>

<title>Welcome to Airplan</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/tbg.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="searchFlightsGui.action?userId=${userId}">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<img src="imgs/warehousesideerr.jpg" />

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

makePayment.ftl

<html>

<head>

<title>Flight Reservation Payment</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="searchFlightsGui.action?userId=${userId}">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<#if userId??>

<div id="corner-rnd" class="col-600 bg-gray">

You have successfully paid for your booking.<br>

A summary of your payment is shown below.<br>

</div>

</#if>

<table id="pay\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Payment ID</th>

<th>Booking ID</th>

<th>Payment Method</th>

<th>Amount</th>

<th>Credit Card Holder</th>

<th>Credit Card Number</th>

</tr>

</thead>

<#if billing?? >

<tbody>

<td>${billing.id}</td>

<td>${billing.bookingId}</td>

<td><#if isPayByPoints == "1">Loyalty Points<#else>Credit Card</#if></td>

<td>${billing.amount}</td>

<td><#if isPayByPoints == "0">${billing.name}<#else>-</#if></td>

<td><#if isPayByPoints == "0">${billing.creditCardNum}<#else>-</#if></td>

</tbody>

<#else>

Billing was not set

</#if>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

makePayment\_gui.ftl

<html>

<head>

<title>Flight Reservation Payment</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function confirmThis() {

var confirmed = confirm("You are already here...\nClicking Okay will clear the current page");

return confirmed;

}

function youSure() {

var waitasec = confirm("If you navigate away from this page,\nyour current bookings will be lost.\nContinue?");

return waitasec;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="" onclick="return confirmThis()">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}" onclick="return youSure()">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}" onclick="return youSure()">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<#if userId??>

<div id="corner-rnd" class="col-600 bg-gray">

Please review your reservation details below.<br>

You may modify your booking(s) after payment has been processed.

</div>

</#if>

<table id="pay\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Flight Number</th>

<th>Passenger Name</th>

<th>Travel Class</th>

<th>Seat Preference</th>

<th>Meal Preference</th>

<th>Fare Price</th>

</tr>

</thead>

<#if ticketList?? >

<#list ticketList as ticket>

<tbody>

<#if ticket.flightId?? && ticket.seatType?? && ticket.seatPref?? && ticket.mealPref?? && ticket.passengerName?? && ticket.cost??>

<tr>

<td>${ticket.flightId}</td>

<td>${ticket.passengerName}</td>

<td><#if ticket.seatType == "1">Business<#else>Economy</#if></td>

<td>${ticket.seatPref}</td>

<td>${ticket.mealPref}</td>

<td>${ticket.cost}</td>

</tr>

</#if>

</tbody>

</#list>

<#else>

Ticket List was not set

</#if>

</table>

<fieldset class="col-600" id="corner-rnd">

<legend>Reservation Details</legend>

<form name="form1" method="get" action="/airplan\_repo/makePayment.action">

<div class="labels-wide">

<p><label for="creditCardName">Credit Card Holder:</label></p>

<p><label for="creditCardNum">Credit Card Number:</label></p>

<p><label for="isPayByPoints">Payment Method:</label></p>

<p><label for="subtotal">Total:</label></p>

</div>

<div class="inputs-wide">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="bookingId" id="bookingId" type="hidden" value=${bookingId} />

<input name="subtotal" id="subtotal" type="hidden" value=${subtotal} />

<p><input name="creditCardName" class="textbox-gen" id="creditCardName" type="text" /></p>

<p><input name="creditCardNum" class="textbox-gen" id="creditCardNum" type="text" /></p>

<p><input type="radio" name="isPayByPoints" checked value="0" /><label for="cc">Credit Card</label>

<#if canPayByPoints == "false">

<input type="radio" disabled /><label for="lp">Loyalty Points</label>

<#else>

<input type="radio" name="isPayByPoints" value="1" /><label for="lp">Loyalty Points(${availPoints} points)</label>

</#if></p>

<p><input name="subtotal\_show" type="text" style="background-color:#fff" value=${subtotal} disabled /></p>

<div style="width:111px"><input type="submit" class="positive btn-gen fl-l" value="Pay" /></div>

</div>

</form>

</fieldset>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

modifyTicket.ftl

<html>

<head>

<title>Update Ticket</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function confirmThis() {

var confirmed = confirm("You are already here...\nClicking Okay will clear the current page");

return confirmed;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*USERNAME HERE\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="searchFlightsGui.action?userId=${userId}">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<#if ticket??>

<table id="row-focus" class="ta-c fs-n">

<thead>

<tr>

<th>ID</th>

<th>Flight ID</th>

<th>Booking ID</th>

<th>Seat Type</th>

<th>Seat Preference</th>

<th>Meal Preference</th>

<th>Passenger Name</th>

</tr>

</thead>

<tbody>

<#if ticket.id?? && ticket.flightId?? && ticket.bookingId?? && ticket.seatType?? && ticket.seatPref?? && ticket.mealPref?? && ticket.passengerName??>

<tr>

<td>${ticket.id}</td>

<td>${ticket.flightId}</td>

<td>${ticket.bookingId}</td>

<td><#if ticket.seatType == "1">Business<#else>Economy</#if></td>

<td>${ticket.seatPref}</td>

<td>${ticket.mealPref}</td>

<td>${ticket.passengerName}</td>

</tr>

</#if>

</tbody>

<#else>

Ticket was not set.

</#if>

</table>

<div class="inputs">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="ticketId" id="ticketId" type="hidden" value=${ticketId} />

</div>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

modifyTicket\_gui.ftl

<html>

<head>

<title>Update Ticket</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function confirmThis() {

var confirmed = confirm("You are already here...\nClicking Okay will clear the current page");

return confirmed;

}

function isEmpty() {

var inputfld\_pname = document.getElementById('passengerName');

if (inputfld\_pname.value == "") {

alert("Please enter the passenger's name");

return false;

}

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*USERNAME HERE\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="searchFlightsGui.action?userId=${userId}">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<fieldset class="col-600" id="corner-rnd">

<legend>Ticket Details</legend>

<form name="form1" method="get" onsubmit="return isEmpty()" action="/airplan\_repo/modifyTicket.action">

<div class="labels">

<p><label for="passengerName">Passenger Name:</label></p>

<p><label for="seatPreference">Seat Preference:</label></p>

<p><label for="mealPreference">Meal Preference:</label></p>

</div>

<div class="inputs">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="ticketId" id="ticketId" type="hidden" value=${ticketId} />

<p><input name="passengerName" class="textbox-gen" id="passengerName" type="text" /></p>

<div style="width:300px">

<input type="radio" name="seatPreference" value="No Preference" checked /><label for="np">No Preference</label>

<input type="radio" name="seatPreference" value="Window Seat" /><label for="window">Window Seat</label>

<input type="radio" name="seatPreference" value="Aisle Seat" /><label for="aisle">Aisle Seat</label>

</div>

<div style="width:300px">

<input type="radio" name="mealPreference" value="No Preference" checked /><label for="np">No Preference</label>

<input type="radio" name="mealPreference" value="Vegetarian" /><label for="veg">Vegetarian</label>

</div>

<input type="submit" class="positive btn-gen" value="Update" />

</div>

</form>

</fieldset>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

newUser.ftl

<html>

<head>

<title>Create New User</title>

<link rel="stylesheet" type="text/css" href="css/airplan\_css.css" />

<script>

function isEmpty() {

var inputfld\_nm = document.getElementById('name');

var inputfld\_eml = document.getElementById('emailAddress');

var inputfld\_pwd = document.getElementById('password');

var inputfld\_pwdc = document.getElementById('passwordCopy');

if (inputfld\_nm.value == "" || inputfld\_eml.value == "" || inputfld\_pwd.value == "" || inputfld\_pwdc.value == "") {

alert("Missing Field(s)");

return false;

}

if (inputfld\_pwd.value != inputfld\_pwdc.value) {

alert("Error: Passwords do not match");

return false;

}

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

<img src="css/pictoptest.png"/>

</div>

</div>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<fieldset class="col-480" id="corner-rnd">

<legend>Create A New User</legend>

<form name="form1" method="get" onsubmit="return isEmpty()" action="/airplan\_repo/createUser.action">

<div class="labels-wide ta-r">

<p><label for="name">Name: &nbsp;</label></p>

<p><label for="emailAddress">Email Address: &nbsp;</label>

<p><label for="password">Password: &nbsp;</label></p>

<p><label for="passwordCopy">Re-enter Password: &nbsp;</label></p>

</div>

<div class="inputs-wide">

<p><input name="name" class="textbox-gen" id="name" type="text" value="" /></p>

<p><input name="emailAddress" class="textbox-gen" id="emailAddress" type="text" value="" /></p>

<p><input name="password" class="textbox-gen" id="password" type="password" value="" /></p>

<p><input name="passwordCopy" class="textbox-gen" id="passwordCopy" type="password" value="" /></p>

<p class="ta-c"><input type="submit" class="positive btn-gen fl-c" value="Create" /></p>

</div>

</fieldset>

</form>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-723">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

reserveFlights.ftl

<html>

<head>

<title>Flight Reservation</title>

<link rel="stylesheet" type="text/css" href="css/airplan\_css.css" />

<script>

function confirmThis() {

var confirmed = confirm("You are already here...\nClicking Okay will clear the current page");

return confirmed;

}

function youSure() {

var waitasec = confirm("If you navigate away from this page,\nyour current bookings will be lost.\nContinue?");

return waitasec;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="" onclick="return confirmThis()">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}" onclick="return youSure()">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}" onclick="return youSure()">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<#if user??>

<div id="corner-rnd" class="col-600 bg-gray">

You have successfully set up your booking options.<br>

Please review your flight settings below, and select Confirm to proceed to payment.<br>

If you wish to add another flight, please select Add Another Flight.

</div>

</#if>

<table id="reserve\_table" class="row-focus ta-c fs-n">

<#if flightList??>

<thead>

<tr>

<th>Passenger Name</th>

<th>Flight Number</th>

<th>Origin</th>

<th>Destination</th>

<th>Departure Time</th>

<th>Arrival Time</th>

<th>Airline</th>

<th>Travel Class</th>

<th>Meal Preference</th>

<th>Seat Preference</th>

<th>Fare Price</th>

</tr>

</thead>

<#list flightList as flight>

<tbody>

<#if user.name?? && passengerName?? && seatType?? && flight.id?? && flight.origin?? && flight.destination?? && flight.departureTime?? && flight.arrivalTime?? && flight.airlineName?? && flight.cost??>

<tr>

<td>${passengerName}</td>

<td>${flight.id}</td>

<td>${flight.originName}</td>

<td>${flight.destinationName}</td>

<td>${flight.departureTime}</td>

<td>${flight.arrivalTime}</td>

<td>${flight.airlineName}</td>

<td><#if seatType == "1">Business<#else>Economy</#if></td>

<td>${mealPreference}</td>

<td>${seatPreference}</td>

<td>${flight.cost}</td>

</tr>

</#if>

</tbody>

</#list>

<#else>

Flight List was not set

</#if>

</table>

<div class="col-600">

<form name="form1" method="get" action="/airplan\_repo/searchFlightsLoopGui.action">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="bookingId" id="bookingId" type="hidden" value=${bookingId} />

<button type="submit" class="positive btn-gen fl-l" name="Submit">Add Another Flight</button>

</form>

<form name="form2" method="get" action="/airplan\_repo/searchFlightsReturnGui.action">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="bookingId" id="bookingId" type="hidden" value=${bookingId} />

<input name="origin" id="origin" type="hidden" value=${destination} />

<input name="destination" id="destination" type="hidden" value=${origin} />

<button type="submit" class="positive btn-gen fl-r" name="Submit">Find Return Flight</button>

</form>

<form name="form3" method="get" action="/airplan\_repo/makePaymentGui.action">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="bookingId" id="bookingId" type="hidden" value=${bookingId} />

<button type="submit" class="positive btn-gen fl-r" name="Submit">Checkout</button>

</form>

</div>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

reserveFlightsLoop.ftl

<html>

<head>

<title>Flight Reservation</title>

<link rel="stylesheet" type="text/css" href="css/airplan\_css.css" />

<script>

function confirmThis() {

var confirmed = confirm("You are already here...\nClicking Okay will clear the current page");

return confirmed;

}

function youSure() {

var waitasec = confirm("If you navigate away from this page,\nyour current bookings will be lost.\nContinue?");

return waitasec;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.ftl">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="" onclick="return confirmThis()">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}" onclick="return youSure()">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}" onclick="return youSure()">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<#if user??>

<div id="corner-rnd" class="col-600 bg-gray">

You have successfully set up your booking options.<br>

Please review your flight settings below, and select Confirm to proceed to payment.<br>

If you wish to add another flight, please select Add Another Flight.

</div>

</#if>

<table id="reserve\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Passenger Name</th>

<th>Flight Number</th>

<th>Origin</th>

<th>Destination</th>

<th>Departure Time</th>

<th>Arrival Time</th>

<th>Airline</th>

<th>Travel Class</th>

<th>Meal Preference</th>

<th>Seat Preference</th>

<th>Fare Price</th>

</tr>

</thead>

<#if flightList?? >

<#list flightList as flight>

<tbody>

<#if user.name?? && passengerName?? && seatType?? && flight.id?? && flight.origin?? && flight.destination?? && flight.departureTime?? && flight.arrivalTime?? && flight.airlineName?? && flight.cost??>

<tr>

<td>${passengerName}</td>

<td>${flight.id}</td>

<td>${flight.originName}</td>

<td>${flight.destinationName}</td>

<td>${flight.departureTime}</td>

<td>${flight.arrivalTime}</td>

<td>${flight.airlineName}</td>

<td><#if seatType == "1">Business<#else>Economy</#if></td>

<td>${mealPreference}</td>

<td>${seatPreference}</td>

<td>${flight.cost}</td>

</tr>

</#if>

</tbody>

</#list>

<#else>

Flight List was not set

</#if>

</table>

<div class="col-600">

<form name="form1" method="get" action="/airplan\_repo/searchFlightsLoopGui.action">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="bookingId" id="bookingId" type="hidden" value=${bookingId} />

<input type="submit" class="positive btn-gen fl-l" value="Add Another Flight" />

</form>

<form name="form2" method="get" action="/airplan\_repo/searchFlightsReturnGui.action">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="bookingId" id="bookingId" type="hidden" value=${bookingId} />

<input name="origin" id="origin" type="hidden" value=${destination} />

<input name="destination" id="destination" type="hidden" value=${origin} />

<input type="submit" class="positive btn-gen fl-r" value="Find Return Flight" />

</form>

<form name="form3" method="get" action="/airplan\_repo/makePaymentGui.action">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="bookingId" id="bookingId" type="hidden" value=${bookingId} />

<input type="submit" class="positive btn-gen fl-r" value="Checkout" />

</form>

</div>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

reserveFlightsLoop\_gui.ftl

<html>

<head>

<title>Reserve Another Flight</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function confirmThis() {

var confirmed = confirm("You are already here...\nClicking Okay will clear the current page");

return confirmed;

}

function youSure() {

var waitasec = confirm("If you navigate away from this page,\nyour current bookings will be lost.\nContinue?");

return waitasec;

}

function isEmpty() {

var inputfld\_pname = document.getElementById('passengerName');

if (inputfld\_pname.value == "") {

alert("Please enter the passenger's name");

return false;

}

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="" onclick="return confirmThis()">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}" onclick="return youSure()">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}" onclick="return youSure()">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<fieldset class="col-600" id="corner-rnd">

<legend>Reservation Details</legend>

<form name="form1" method="get" onsubmit="return isEmpty()" action="/airplan\_repo/reserveFlightsLoop.action">

<div class="labels">

<p><label for="passengerName">Passenger Name:</label></p>

<p><label for="seatPreference">Seat Preference:</label></p>

<p><label for="mealPreference">Meal Preference:</label></p>

</div>

<div class="inputs">

<input name="flightId" id="flightId" type="hidden" value=${flightId} />

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="bookingId" id="bookingId" type="hidden" value=${bookingId} />

<input name="seatType" id="seatType" type="hidden" value=${seatType} />

<input name="passengerName" class="textbox-gen" id="passengerName" type="text" />

<div style="width:300px">

<input type="radio" name="seatPreference" value="No Preference" checked /><label for="np">No Preference</label>

<input type="radio" name="seatPreference" value="Window Seat" /><label for="window">Window Seat</label>

<input type="radio" name="seatPreference" value="Aisle Seat" /><label for="aisle">Aisle Seat</label>

</div>

<div style="width:300px">

<input type="radio" name="mealPreference" value="No Preference" checked /><label for="np">No Preference</label>

<input type="radio" name="mealPreference" value="Vegetarian" /><label for="veg">Vegetarian</label>

</div>

<input type="submit" class="positive btn-gen" value="Reserve" />

</div>

</form>

</fieldset>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

reserveFlights\_gui.ftl

<html>

<head>

<title>Reserve a Flight</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function confirmThis() {

var confirmed = confirm("You are already here...\nClicking Okay will clear the current page");

return confirmed;

}

function youSure() {

var waitasec = confirm("If you navigate away from this page,\nyour current inputs will be lost.\nContinue?");

return waitasec;

}

function isEmpty() {

var inputfld\_pname = document.getElementById('passengerName');

if (inputfld\_pname.value == "") {

alert("Please enter the passenger's name");

return false;

}

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="" onclick="return confirmThis()">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}" onclick="return youSure()">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}" onclick="return youSure()">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<fieldset class="col-600" id="corner-rnd">

<legend>Reservation Details</legend>

<form name="form1" method="get" onsubmit="return isEmpty()" action="/airplan\_repo/reserveFlights.action">

<div class="labels">

<p><label for="passengerName">Passenger Name:</label></p>

<p><label for="seatPreference">Seat Preference:</label></p>

<p><label for="mealPreference">Meal Preference:</label></p>

</div>

<div class="inputs">

<input name="flightId" id="flightId" type="hidden" value=${flightId} />

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="seatType" id="seatType" type="hidden" value=${seatType} />

<input name="passengerName" class="textbox-gen" id="passengerName" type="text" />

<div style="width:300px">

<input type="radio" name="seatPreference" value="No Preference" checked /><label for="np">No Preference</label>

<input type="radio" name="seatPreference" value="Window Seat" /><label for="window">Window Seat</label>

<input type="radio" name="seatPreference" value="Aisle Seat" /><label for="aisle">Aisle Seat</label>

</div>

<div style="width:300px">

<input type="radio" name="mealPreference" value="No Preference" checked /><label for="np">No Preference</label>

<input type="radio" name="mealPreference" value="Vegetarian" /><label for="veg">Vegetarian</label>

</div>

<input type="submit" class="positive btn-gen" value="Reserve" />

</div>

</form>

</fieldset>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

searchFlights.ftl

<html>

<head>

<title>Search for Flights</title>

<link rel="stylesheet" type="text/css" href="css/airplan\_css.css" />

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<table id="search\_table" class="row-focus-fancy ta-c fs-n">

<#if flightList??>

<#if (flightList)?has\_content>

<thead>

<tr>

<th>Flight Number</th>

<th>Origin</th>

<th>Destination</th>

<th>Departure Time</th>

<th>Arrival Time</th>

<th>Airline</th>

<th>Available Seats</th>

<th>Fare Price</th>

</tr>

</thead>

<#list flightList as flight>

<tbody>

<#if flight.id?? && flight.origin?? && flight.destination?? && flight.departureTime?? && flight.arrivalTime?? && flight.airlineName?? && flight.availableSeats?? && flight.cost??>

<tr>

<td onclick="window.location.href='reserveFlightsGui.action?userId=${userId}&flightId=${flight.id}&seatType=${seatType}'">${flight.id}</td>

<td onclick="window.location.href='reserveFlightsGui.action?userId=${userId}&flightId=${flight.id}&seatType=${seatType}'">${flight.originName}</td>

<td onclick="window.location.href='reserveFlightsGui.action?userId=${userId}&flightId=${flight.id}&seatType=${seatType}'">${flight.destinationName}</td>

<td onclick="window.location.href='reserveFlightsGui.action?userId=${userId}&flightId=${flight.id}&seatType=${seatType}'">${flight.departureTime}</td>

<td onclick="window.location.href='reserveFlightsGui.action?userId=${userId}&flightId=${flight.id}&seatType=${seatType}'">${flight.arrivalTime}</td>

<td onclick="window.location.href='reserveFlightsGui.action?userId=${userId}&flightId=${flight.id}&seatType=${seatType}'">${flight.airlineName}</td>

<td onclick="window.location.href='reserveFlightsGui.action?userId=${userId}&flightId=${flight.id}&seatType=${seatType}'">${flight.availableSeats}</td>

<td onclick="window.location.href='reserveFlightsGui.action?userId=${userId}&flightId=${flight.id}&seatType=${seatType}'">${flight.cost}</td>

</tr>

</#if>

</tbody>

</#list>

<#else>

<div class="ta-c">

<div id="corner-rnd" class="col-600 bg-gray">No flights matching set criteria on selected date.</div>

<div class="p-basic"></div>

<a href="javascript:history.back()"><button type="submit" class="positive btn-gen fl-c">Search Again</button></a>

</div>

</#if>

<#else>

Flight List was not set

</#if>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

searchFlightsLoop.ftl

<html>

<head>

<title>Search for Flights</title>

<link rel="stylesheet" type="text/css" href="css/airplan\_css.css" />

<script>

function youSure() {

var waitasec = confirm("If you navigate away from this page,\nyour current bookings will be lost.\nContinue?");

return waitasec;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}" onclick="return youSure()">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}" onclick="return youSure()">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<table id="search\_table" class="row-focus-fancy ta-c fs-n">

<#if flightList??>

<#if (flightList)?has\_content>

<thead>

<tr>

<th>Flight Number</th>

<th>Origin</th>

<th>Destination</th>

<th>Departure Time</th>

<th>Arrival Time</th>

<th>Airline</th>

<th>Available Seats</th>

<th>Fare Price</th>

</tr>

</thead>

<#list flightList as flight>

<tbody>

<#if flight.id?? && flight.origin?? && flight.destination?? && flight.departureTime?? && flight.arrivalTime?? && flight.airlineName?? && flight.availableSeats?? && flight.cost??>

<tr>

<td onclick="window.location.href='reserveFlightsLoopGui.action?userId=${userId}&bookingId=${bookingId}&flightId=${flight.id}&seatType=${seatType}'">${flight.id}</td>

<td onclick="window.location.href='reserveFlightsLoopGui.action?userId=${userId}&bookingId=${bookingId}&flightId=${flight.id}&seatType=${seatType}'">${flight.originName}</td>

<td onclick="window.location.href='reserveFlightsLoopGui.action?userId=${userId}&bookingId=${bookingId}&flightId=${flight.id}&seatType=${seatType}'">${flight.destinationName}</td>

<td onclick="window.location.href='reserveFlightsLoopGui.action?userId=${userId}&bookingId=${bookingId}&flightId=${flight.id}&seatType=${seatType}'">${flight.departureTime}</td>

<td onclick="window.location.href='reserveFlightsLoopGui.action?userId=${userId}&bookingId=${bookingId}&flightId=${flight.id}&seatType=${seatType}'">${flight.arrivalTime}</td>

<td onclick="window.location.href='reserveFlightsLoopGui.action?userId=${userId}&bookingId=${bookingId}&flightId=${flight.id}&seatType=${seatType}'">${flight.airlineName}</td>

<td onclick="window.location.href='reserveFlightsLoopGui.action?userId=${userId}&bookingId=${bookingId}&flightId=${flight.id}&seatType=${seatType}'">${flight.availableSeats}</td>

<td onclick="window.location.href='reserveFlightsLoopGui.action?userId=${userId}&bookingId=${bookingId}&flightId=${flight.id}&seatType=${seatType}'">${flight.cost}</td>

</tr>

</#if>

</tbody>

</#list>

<#else>

<div class="ta-c">

<div id="corner-rnd" class="col-600 bg-gray">No flights matching set criteria for selected date.</div>

<div class="p-basic"></div>

<a href="javascript:history.back()"><button type="submit" class="positive btn-gen fl-c">Search Again</button></a>

</div>

</#if>

<#else>

Flight List was not set

</#if>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

searchFlightsLoop\_gui.ftl

<html>

<head>

<title>Search for More Flights</title>

<link rel="stylesheet" href="css/jquery-ui.css" />

<script src="js/jquery-1.8.2.js"></script>

<script src="js/jquery-ui.js"></script>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

$(function() {

var places = [

"Atlanta","Bangkok","Beijing","Calgary","Chicago","Edmonton","Frankfurt","Hong Kong","Houston","Kelowna",

"London","Los Angeles","Madrid","Montreal","New York","Orlando","Osaka","Ottawa","Paris","Phoenix","San Francisco",

"Saskatoon","Seattle","Shanghai","Singapore","Tokyo","Toronto","Vancouver","Whitehorse","Winnipeg","Yellowknife"

];

$( ".datepicker" ).datepicker({dateFormat: "yy-mm-dd"});

$( ".places" ).autocomplete({source: places});

});

function confirmThis() {

var confirmed = confirm("You are already here...\nClicking Okay will clear the current page");

return confirmed;

}

function youSure() {

var waitasec = confirm("If you navigate away from this page,\nyour current bookings will be lost.\nContinue?");

return waitasec;

}

function isEmpty() {

var inputfld\_dest = document.getElementById('destination');

var inputfld\_orig = document.getElementById('origin');

var inputfld\_dtime = document.getElementById('departureTime');

var inputfld\_reqst = document.getElementById('requiredSeats');

if (inputfld\_dest.value == "" || inputfld\_orig.value == "" || inputfld\_dtime.value == "" || inputfld\_reqst.value == "") {

alert("Missing Field(s)");

return false;

}

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="" onclick="return confirmThis()">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}" onclick="return youSure()">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions?userId=${userId}" onclick="return youSure()">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<fieldset class="col-600" id="corner-rnd">

<legend>Search for Another Flight</legend>

<form name="form1" method="get" onsubmit="return isEmpty()" action="/airplan\_repo/searchFlightsLoop.action">

<div class="labels">

<p><label for="destination">Destination:</label></p>

<p><label for="origin">Origin:</label></p>

<p><label for="departureTime">Departure Date:</label></p>

<p><label for="requiredSeats">Required Seats:</label></p>

<p><label for="seatType">Seat Type:</label></p>

</div>

<div class="inputs">

<input name="bookingId" id="bookingId" type="hidden" value=${bookingId} />

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="destination" class="textbox-gen places" id="destination" type="text" />

<input name="origin" class="textbox-gen places" id="origin" type="text" />

<input name="departureTime" class="textbox-gen datepicker" id="departureTime" type="text" />

<input name="requiredSeats" class="textbox-gen" id="requiredSeats" type="number" />

<div>

<input type="radio" name="seatType" value="2" checked /><label for="2">Economy</label>

<input type="radio" name="seatType" value="1" /><label for="1">Business</label>

</div>

<input type="submit" class="positive btn-gen" value="Search" />

</div>

<div><img class="fl-r" src="css/map-icon.png" /></div>

<form>

</fieldset>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

searchFlightsReturn\_gui.ftl

<html>

<head>

<title>Search for More Flights</title>

<link rel="stylesheet" href="css/jquery-ui.css" />

<script src="js/jquery-1.8.2.js"></script>

<script src="js/jquery-ui.js"></script>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

$(function() {

var places = [

"Atlanta","Bangkok","Beijing","Calgary","Chicago","Edmonton","Frankfurt","Hong Kong","Houston","Kelowna",

"London","Los Angeles","Madrid","Montreal","New York","Orlando","Osaka","Ottawa","Paris","Phoenix","San Francisco",

"Saskatoon","Seattle","Shanghai","Singapore","Tokyo","Toronto","Vancouver","Whitehorse","Winnipeg","Yellowknife"

];

$( ".datepicker" ).datepicker({dateFormat: "yy-mm-dd"});

$( ".places" ).autocomplete({source: places});

});

function confirmThis() {

var confirmed = confirm("You are already here...\nClicking Okay will clear the current page");

return confirmed;

}

function youSure() {

var waitasec = confirm("If you navigate away from this page,\nyour current bookings will be lost.\nContinue?");

return waitasec;

}

function isEmpty() {

var inputfld\_dest = document.getElementById('destination');

var inputfld\_orig = document.getElementById('origin');

var inputfld\_dtime = document.getElementById('departureTime');

var inputfld\_reqst = document.getElementById('requiredSeats');

if (inputfld\_dest.value == "" || inputfld\_orig.value == "" || inputfld\_dtime.value == "" || inputfld\_reqst.value == "") {

alert("Missing Field(s)");

return false;

}

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="" onclick="return confirmThis()">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}" onclick="return youSure()">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}" onclick="return youSure()">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<fieldset class="col-600" id="corner-rnd">

<legend>Search for Return Flight</legend>

<form name="form1" method="get" onsubmit="return isEmpty()" action="/airplan\_repo/searchFlightsLoop.action">

<div class="labels">

<p><label for="destination">Destination:</label></p>

<p><label for="origin">Origin:</label></p>

<p><label for="departureTime">Departure Date:</label></p>

<p><label for="requiredSeats">Required Seats:</label></p>

<p><label for="seatType">Seat Type:</label></p>

</div>

<div class="inputs">

<input name="bookingId" id="bookingId" type="hidden" value=${bookingId} />

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="destination" class="textbox-gen places" id="destination" value=${destination} type="text" />

<input name="origin" class="textbox-gen places" id="origin" value=${origin} type="text" />

<input name="departureTime" class="textbox-gen datepicker" id="departureTime" type="text" />

<input name="requiredSeats" class="textbox-gen" id="requiredSeats" type="number" />

<div>

<input type="radio" name="seatType" value="2" checked /><label for="2">Economy</label>

<input type="radio" name="seatType" value="1" /><label for="1">Business</label>

</div>

<input type="submit" class="positive btn-gen" value="Search" />

</div>

<div><img class="fl-r" src="css/map-icon.png" /></div>

<form>

</fieldset>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

searchFlights\_gui.ftl

<html>

<head>

<title>Search for Flights</title>

<link rel="stylesheet" href="css/jquery-ui.css" />

<script src="js/jquery-1.8.2.js"></script>

<script src="js/jquery-ui.js"></script>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

$(function() {

var places = [

"Atlanta","Bangkok","Beijing","Calgary","Chicago","Edmonton","Frankfurt","Hong Kong","Houston","Kelowna",

"London","Los Angeles","Madrid","Montreal","New York","Orlando","Osaka","Ottawa","Paris","Phoenix","San Francisco",

"Saskatoon","Seattle","Shanghai","Singapore","Tokyo","Toronto","Vancouver","Whitehorse","Winnipeg","Yellowknife"

];

$( ".datepicker" ).datepicker({dateFormat: "yy-mm-dd"});

$( ".places" ).autocomplete({source: places});

});

function confirmThis() {

return false;

}

function youSure() {

var waitasec = confirm("If you navigate away from this page,\nyour current bookings will be lost.\nContinue?");

return waitasec;

}

function isEmpty() {

var inputfld\_dest = document.getElementById('destination');

var inputfld\_orig = document.getElementById('origin');

var inputfld\_dtime = document.getElementById('departureTime');

var inputfld\_reqst = document.getElementById('requiredSeats');

if (inputfld\_dest.value == "" || inputfld\_orig.value == "" || inputfld\_dtime.value == "" || inputfld\_reqst.value == "") {

alert("Missing Field(s)");

return false;

}

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="" onclick="return confirmThis()">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<fieldset class="col-600" id="corner-rnd">

<legend>Search for a Flight</legend>

<form name="form1" method="get" onsubmit="return isEmpty()" action="/airplan\_repo/searchFlights.action">

<div class="labels">

<p><label for="destination">Destination:</label></p>

<p><label for="origin">Origin:</label></p>

<p><label for="departureTime">Departure Date:</label></p>

<p><label for="requiredSeats">Required Seats:</label></p>

<p><label for="seatType">Seat Type:</label></p>

</div>

<div class="inputs">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="destination" class="textbox-gen places" id="destination" type="text" />

<input name="origin" class="textbox-gen places" id="origin" type="text" />

<input name="departureTime" class="textbox-gen datepicker" id="departureTime" type="text" />

<input name="requiredSeats" class="textbox-gen" id="requiredSeats" type="number" />

<div>

<input type="radio" name="seatType" value="2" checked /><label for="2">Economy</label>

<input type="radio" name="seatType" value="1" /><label for="1">Business</label>

</div>

<input type="submit" class="positive btn-gen" value="Search" />

</div>

<div><img class="fl-r" src="css/map-icon.png" /></div>

<form>

</fieldset>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

selectManifest.ftl

<html>

<head>

<title>View Flight Manifest</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script src="js/jquery-1.8.2.js"/></script>

<script src="js/jquery.tablescroll.js"></script>

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/2.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="staffSignIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="" onclick="return notDone()">Manage User Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="selectManifest.action?userId=${userId}">View Flight Manifests</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewStatistics.action?userId=${userId}">View Flight Statistics</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<div id="corner-rnd" class="col-600 bg-gray">

${message}

</div>

<table id="flights" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Flight ID</th>

<th>Origin</th>

<th>Destination</th>

<th>Departure Time</th>

<th>Arrival Time</th>

<th>Airline</th>

</tr>

</thead>

<#if selectionList??>

<tbody>

<#list selectionList as selection>

<tr>

<td><a href="viewManifest.action?flightManifest\_flightId=${selection.id}">${selection.id}</a></td>

<td>${selection.origin}</td>

<td>${selection.destination}</td>

<td>${selection.departureTime}</td>

<td>${selection.arrivalTime}</td>

<td>${selection.airlineName}</td>

</tr>

</#list>

</tbody>

<#else>

Error!

</#if>

</table>

<script>

jQuery(document).ready(function($) {

$('#flights').tableScroll({height:350});

});

</script>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

signIn.ftl

<html>

<head>

<title>Please Log In</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function isEmpty() {

var inputfld\_eml = document.getElementById('emailAddress');

var inputfld\_pwd = document.getElementById('password');

if (inputfld\_eml.value == "" || inputfld\_pwd.value == "") {

alert("Missing Field(s)");

return false;

}

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

</div>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<fieldset class="col-480" id="corner-rnd">

<legend>Login</legend>

<form name="form1" method="get" onsubmit="return isEmpty()" action="login.action">

<div class="labels-wide ta-r">

<p><label for="emailAddress">Email: &nbsp;</label>

<p><label for="password">Password: &nbsp;</label></p>

</div>

<div class="inputs-wide">

<p><input name="emailAddress" class="textbox-gen" id="emailAddress" type="text" value="" /></p>

<p><input name="password" class="textbox-gen" id="password" type="password" value="" /></p>

<p class="ta-c">

<input type="submit" class="positive btn-gen fl-c" value="Login" /> | <a href="staffSignIn.action">Administrator?</a>

<br><a href="createUserInput.action">I am a new customer...</a>

</p>

</div>

</form>

</fieldset>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

staffCancelBooking.ftl

<html>

<head>

<title>Staff Cancel Booking</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="" onclick="return notDone()">Manage User Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="selectManifest.action?userId=${userId}">View Flight Manifests</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewStatistics.action?userId=${userId}">View Flight Statistics</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<#if cancelState??>

<#if cancelState==0>

<div id="corner-rnd" class="col-600 bg-gray">Billing summary of selected booking:</div>

<table id="b\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Bill ID</th>

<th>Date Created</th>

<th>Amount</th>

<th>Name</th>

<th>Credit Card</th>

<th>Points Used?</th>

</tr>

</thead>

<#if bill??>

<tbody>

<#if bill.id?? && bill.dateCreated?? && bill.amount?? && bill.isLoyaltyPointsUsed??>

<tr>

<td>${bill.id}</td>

<td>${bill.dateCreated}</td>

<td>${bill.amount}</td>

<<td><#if bill.name??>${bill.name}<#else>-</#if></td>

<td><#if bill.creditCardNum??>${bill.creditCardNum}<#else>-</#if></td>

<td>${bill.isLoyaltyPointsUsed()?string("yes", "no")}</td>

</tr>

</#if>

</tbody>

<#else>

Bill was not set

</#if>

</table>

<div id="corner-rnd" class="col-600 bg-gray">This booking contains the following reservations:</div>

<table id="r\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Ticket ID</th>

<th>Flight ID</th>

<th>Seat Type</th>

<th>Seat Preference</th>

<th>Meal Preference</th>

<th>Passenger Name</th>

</tr>

</thead>

<#if ticketList??>

<#list ticketList as ticket>

<tbody>

<#if ticket.id?? && ticket.flightId?? && ticket.seatType?? && ticket.seatPref?? && ticket.mealPref?? && ticket.passengerName??>

<tr>

<td>${ticket.id}</td>

<td>${ticket.flightId}</td>

<td>${ticket.seatType}</td>

<td>${ticket.seatPref}</td>

<td>${ticket.mealPref}</td>

<td>${ticket.passengerName}</td>

</tr>

</#if>

</tbody>

</#list>

<#else>

Tickets were not set

</#if>

</table>

<div class="p-basic col-768">

<a href="staffConfirmCancel.action?userId=${userId}&bookingId=${bookingId}&billingId=${bill.id}&customerId=${customerId}"><button type="submit" class="positive btn-gen fl-r">Confirm Cancellation</button></a>

</div>

</#if>

</#if>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

staffConfirmCancel.ftl

<html>

<head>

<title>Staff Confirm Booking Cancellation</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="" onclick="return notDone()">Manage User Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="selectManifest.action?userId=${userId}">View Flight Manifests</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewStatistics.action?userId=${userId}">View Flight Statistics</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

The booking has been cancelled.

<#if cancelBooking == 1 && cancelTickets != 0 && refundBill == 1>

<p>Booking ${bookingId} has been cancelled.</p>

<p>Bill ${billingId} has been refunded.</p>

<p>${cancelTickets} tickets have been cancelled.</p>

<#else>

There was an error canceling this booking.

</#if>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

staffModifyTicket.ftl

<html>

<head>

<title>Staff Update Ticket</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

function confirmThis() {

var confirmed = confirm("You are already here...\nClicking Okay will clear the current page");

return confirmed;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*USERNAME HERE\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="" onclick="return notDone()">Manage User Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="selectManifest.action?userId=${userId}">View Flight Manifests</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewStatistics.action?userId=${userId}">View Flight Statistics</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<#if ticket??>

<table id="row-focus" class="ta-c fs-n">

<thead>

<tr>

<th>ID</th>

<th>Flight ID</th>

<th>Booking ID</th>

<th>Seat Type</th>

<th>Seat Preference</th>

<th>Meal Preference</th>

<th>Passenger Name</th>

</tr>

</thead>

<tbody>

<#if ticket.id?? && ticket.flightId?? && ticket.bookingId?? && ticket.seatType?? && ticket.seatPref?? && ticket.mealPref?? && ticket.passengerName??>

<tr>

<td>${ticket.id}</td>

<td>${ticket.flightId}</td>

<td>${ticket.bookingId}</td>

<td><#if ticket.seatType == "1">Business<#else>Economy</#if></td>

<td>${ticket.seatPref}</td>

<td>${ticket.mealPref}</td>

<td>${ticket.passengerName}</td>

</tr>

</#if>

</tbody>

<#else>

Ticket was not set.

</#if>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

staffModifyTicket\_gui.ftl

<html>

<head>

<title>Staff Update Ticket</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

function confirmThis() {

var confirmed = confirm("You are already here...\nClicking Okay will clear the current page");

return confirmed;

}

function isEmpty() {

var inputfld\_pname = document.getElementById('passengerName');

if (inputfld\_pname.value == "") {

alert("Please enter the passenger's name");

return false;

}

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<a href="test\_template.html" title="AirPlan">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</a>

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*USERNAME HERE\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="" onclick="return notDone()">Manage User Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="selectManifest.action?userId=${userId}">View Flight Manifests</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewStatistics.action?userId=${userId}">View Flight Statistics</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<fieldset class="col-600" id="corner-rnd">

<legend>Ticket Details</legend>

<form name="form1" method="get" onsubmit="return isEmpty()" action="/airplan\_repo/staffModifyTicket.action">

<div class="labels">

<p><label for="passengerName">Passenger Name:</label></p>

<p><label for="seatPreference">Seat Preference:</label></p>

<p><label for="mealPreference">Meal Preference:</label></p>

</div>

<div class="inputs">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="customerId" id="customerId" type="hidden" value=${customerId} />

<input name="ticketId" id="ticketId" type="hidden" value=${ticketId} />

<p><input name="passengerName" class="textbox-gen" id="passengerName" type="text" /></p>

<div style="width:300px">

<input type="radio" name="seatPreference" value="No Preference" checked /><label for="np">No Preference</label>

<input type="radio" name="seatPreference" value="Window Seat" /><label for="window">Window Seat</label>

<input type="radio" name="seatPreference" value="Aisle Seat" /><label for="aisle">Aisle Seat</label>

</div>

<div style="width:300px">

<input type="radio" name="mealPreference" value="No Preference" checked /><label for="np">No Preference</label>

<input type="radio" name="mealPreference" value="Vegetarian" /><label for="veg">Vegetarian</label>

</div>

<input type="submit" class="positive btn-gen" value="Update" />

</div>

</form>

</fieldset>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

staffSignIn.ftl

<html>

<head>

<title>Administrator Log In</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function isEmpty() {

var inputfld\_eml = document.getElementById('emailAddress');

var inputfld\_pwd = document.getElementById('password');

if (inputfld\_eml.value == "" || inputfld\_pwd.value == "") {

alert("Missing Field(s)");

return false;

}

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

</div>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<fieldset class="col-480" id="corner-rnd">

<legend>Login</legend>

<form name="form1" method="get" onsubmit="return isEmpty()" action="stafflogin.action">

<div class="labels-wide ta-r">

<p><label for="emailAddress">Email: &nbsp;</label>

<p><label for="password">Password: &nbsp;</label></p>

</div>

<div class="inputs-wide">

<p><input name="emailAddress" class="textbox-gen" id="emailAddress" type="text" value="" /></p>

<p><input name="password" class="textbox-gen" id="password" type="password" value="" /></p>

<p class="ta-c">

<input type="submit" class="positive btn-gen fl-c" value="Login" /> | <a href="signIn.action">Not Administrator?</a>

</p>

</div>

</form>

</fieldset>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

staffViewBooking.ftl

<html>

<head>

<title>Staff View Bookings</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="" onclick="return notDone()">Manage User Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="selectManifest.action?userId=${userId}">View Flight Manifests</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewStatistics.action?userId=${userId}">View Flight Statistics</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<#if booking??>

<table id="b\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Booking ID</th>

<th>Date Created</th>

<th>Cancel Booking</th>

</tr>

</thead>

<tbody>

<#if booking?? && booking.id??>

<tr>

<td>${booking.id}</td></a>

<#if booking.dateCreated??>

<td>${booking.dateCreated?string("EEE, MMM d, yyyy")}</td>

</#if>

<#if booking.cancelState == 1>

<td><a href="/airplan\_repo/staffCancelBooking.action?userId=${userId}&bookingId=${booking.id}&customerId=${customerId}">Cancel Booking ${booking.id}</a></td>

<#elseif booking.cancelState == 0>

<td>Cannot Cancel</td>

<#else>

<td>Cancelled</td>

</#if>

</tr>

</#if>

</tbody>

</table>

<#else>

Bookings Were Not Displayed

</#if>

<table id="t\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Ticket ID</th>

<th>Flight ID</th>

<th>Seat Type</th>

<th>Seat Preference</th>

<th>Meal Preference</th>

<th>Passenger Name</th>

<th>Cost</th>

<th>Origin</th>

<th>Destination</th>

<th>Departure Time</th>

<th>Arrival Time</th>

<th>Airline Name</th>

<th>Manage Ticket</th>

</tr>

</thead>

<#if ticketSetList??>

<#list ticketSetList as ts>

<tbody>

<#if ts.ticket.id?? && ts.ticket.flightId?? && ts.ticket.seatType?? && ts.ticket.seatPref?? && ts.ticket.mealPref?? && ts.ticket.passengerName?? && ts.ticket.state?? && ts.ticket.cost?? && ts.flight.origin?? && ts.flight.destination?? && ts.flight.departureTime?? && ts.flight.arrivalTime?? && ts.flight.airlineName??>

<tr>

<td>${ts.ticket.id}</td>

<td>${ts.ticket.flightId}</td>

<td><#if ts.ticket.seatType == "1">Business<#else>Economy</#if></td>

<td>${ts.ticket.seatPref}</td>

<td>${ts.ticket.mealPref}</td>

<td>${ts.ticket.passengerName}</td>

<td>${ts.ticket.cost}</td>

<td>${ts.flight.origin}</td>

<td>${ts.flight.destination}</td>

<td>${ts.flight.departureTime}</td>

<td>${ts.flight.arrivalTime}</td>

<td>${ts.flight.airlineName}</td>

<#if ts.ticket.state == 1>

<td><a href="/airplan\_repo/staffModifyTicketGui.action?ticketId=${ts.ticket.id}&customerId=${customerId}&userId=${userId}">Modify Ticket</a></td>

<#elseif ts.ticket.state == 2>

<td>Cannot Manage Ticket</td>

<#elseif ts.ticket.state == 3>

<td>Ticket Cancelled</td>

</#if>

</tr>

</#if>

</tbody>

</#list>

<#else>

Tickets were not set

</#if>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

staffViewBookingList.ftl

<html>

<head>

<title>Staff View Bookings</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/1.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="" onclick="return notDone()">Manage User Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="selectManifest.action?userId=${userId}">View Flight Manifests</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewStatistics.action?userId=${userId}">View Flight Statistics</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<table id="b\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Booking ID</th>

<th>Date Created</th>

<th>Status</th>

</tr>

</thead>

<#list bookingList as booking>

<tbody>

<#if booking.id?? && booking.dateCreated?? && booking.cancelState??>

<tr>

<td><a href="/airplan\_repo/staffViewBooking.action?userId=${userId}&bookingId=${booking.id}&customerId=${customerId}">${booking.id}</a></td>

<#if booking.dateCreated??>

<td>${booking.dateCreated?string("EEE, MMM d, yyyy")}</td>

</#if>

<#if booking.cancelState == 1>

<td>Valid Booking</td>

<#elseif booking.cancelState == 0>

<td>Boarding Pass Printed</td>

<#else>

<td>Cancelled</td>

</#if>

</tr>

</#if>

</#list>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

transactions.ftl

<html>

<head>

<title>View Past Transactions</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/3.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="searchFlightsGui.action?userId=${userId}">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<table id="txn\_table" class="row-focus-fancy ta-c fs-n">

<#if transactionList??>

<thead>

<tr>

<th>Date</th>

<th>Billing ID</th>

<th>Credit Card Number</th>

<th>Name</th>

<th>Amount</th>

<th>Points Awarded</th>

<th>Reason Points Updated</th>

</tr>

</thead>

<#list transactionList as transaction>

<tbody>

<tr>

<td>${transaction.dateCreated?substring(0,12)}</td>

<td>${transaction.billingId}</td>

<#if transaction.creditCardNumber??>

<td>${transaction.creditCardNumber}</td>

<td>${transaction.name}</td>

<td>$ ${transaction.amount}</td>

<#else>

<td>N/A</td>

<td>N/A</td>

<td>${transaction.amount} Points</td>

</#if>

<#if transaction.pointsAwarded??>

<td>${transaction.pointsAwarded}</td>

<#else>

<td>-</td>

</#if>

<#if transaction.reason??>

<td>${transaction.reason}</td>

<#else>

<td>Points will be awarded after boarding pass is printed.</td>

</#if>

</tr>

</tbody>

</#list>

</table>

<#else>

Transactions not defined

</#if>

<div class="p-basic"></div>

<#if points = -1>

Points Not Set For User

<#else>

Total Points: ${points}

</#if>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

viewBooking.ftl

<html>

<head>

<title>View Bookings</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/2.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="searchFlightsGui.action?userId=${userId}">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<#if booking??>

<table id="b\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Booking ID</th>

<th>Date Created</th>

<th>Print Boarding Pass</th>

<th>Cancel Booking</th>

</tr>

</thead>

<tbody>

<#if booking?? && booking.id??>

<tr>

<td>${booking.id}</td></a>

<#if booking.dateCreated??>

<td>${booking.dateCreated?string("EEE, MMM d, yyyy")}</td>

</#if>

<#if booking.cancelState == 1>

<td><a href="/airplan\_repo/boardingPass.action?userId=${userId}&bookingId=${booking.id}">Print Boarding Pass</a></td>

<td><a href="/airplan\_repo/cancelBooking.action?userId=${userId}&bookingId=${booking.id}">Cancel Booking ${booking.id}</a></td>

<#elseif booking.cancelState == 0>

<td><a href="/airplan\_repo/boardingPass.action?userId=${userId}&bookingId=${booking.id}">Re-Print Boarding Pass</a></td>

<td>Cannot Cancel</td>

<#else>

<td>No Boarding Pass Available</td>

<td>Cancelled</td>

</#if>

</tr>

</#if>

</tbody>

</table>

<#else>

Bookings Were Not Displayed

</#if>

<table id="t\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Ticket ID</th>

<th>Flight ID</th>

<th>Seat Type</th>

<th>Seat Preference</th>

<th>Meal Preference</th>

<th>Passenger Name</th>

<th>Cost</th>

<th>Origin</th>

<th>Destination</th>

<th>Departure Time</th>

<th>Arrival Time</th>

<th>Airline Name</th>

<th>Manage Ticket</th>

</tr>

</thead>

<#if ticketSetList??>

<#list ticketSetList as ts>

<tbody>

<#if ts.ticket.id?? && ts.ticket.flightId?? && ts.ticket.seatType?? && ts.ticket.seatPref?? && ts.ticket.mealPref?? && ts.ticket.passengerName?? && ts.ticket.state?? && ts.ticket.cost?? && ts.flight.origin?? && ts.flight.destination?? && ts.flight.departureTime?? && ts.flight.arrivalTime?? && ts.flight.airlineName??>

<tr>

<td>${ts.ticket.id}</td>

<td>${ts.ticket.flightId}</td>

<td><#if ts.ticket.seatType == "1">Business<#else>Economy</#if></td>

<td>${ts.ticket.seatPref}</td>

<td>${ts.ticket.mealPref}</td>

<td>${ts.ticket.passengerName}</td>

<td>${ts.ticket.cost}</td>

<td>${ts.flight.origin}</td>

<td>${ts.flight.destination}</td>

<td>${ts.flight.departureTime}</td>

<td>${ts.flight.arrivalTime}</td>

<td>${ts.flight.airlineName}</td>

<#if ts.ticket.state == 1>

<td><a href="/airplan\_repo/modifyTicketGui.action?ticketId=${ts.ticket.id}&userId=${userId}">Modify Ticket</a></td>

<#elseif ts.ticket.state == 2>

<td>Cannot Manage Ticket</td>

<#elseif ts.ticket.state == 3>

<td>Ticket Cancelled</td>

</#if>

</tr>

</#if>

</tbody>

</#list>

<#else>

Tickets were not set

</#if>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

viewBookingList.ftl

<html>

<head>

<title>View Bookings</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/2.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="signIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="searchFlightsGui.action?userId=${userId}">Search / Reserve Flights</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="viewBookingList.action?userId=${userId}">Manage Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="viewTransactions.action?userId=${userId}">View History</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<table id="b\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Booking ID</th>

<th>Date Created</th>

<th>Status</th>

</tr>

</thead>

<#list bookingList as booking>

<tbody>

<#if booking.id?? && booking.dateCreated?? && booking.cancelState??>

<tr>

<td><a href="/airplan\_repo/viewBooking.action?userId=${userId}&bookingId=${booking.id}">${booking.id}</a></td>

<#if booking.dateCreated??>

<td>${booking.dateCreated?string("EEE, MMM d, yyyy")}</td>

</#if>

<#if booking.cancelState == 1>

<td>Valid Booking</td>

<#elseif booking.cancelState == 0>

<td>Boarding Pass Printed</td>

<#else>

<td>Cancelled</td>

</#if>

</tr>

</#if>

</#list>

</table>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

viewStatistics.ftl

<html>

<head>

<title>Flight Statistics</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script src="js/jquery-1.8.2.js"/></script>

<script src="js/jquery.tablescroll.js"></script>

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/3.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*${user.name}\* &nbsp; &nbsp; <a class="mm-item-link c9" href="staffSignIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="" onclick="return notDone()">Manage User Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="selectManifest.action?userId=${userId}">View Flight Manifests</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="viewStatistics.action?userId=${userId}">View Flight Statistics</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<#if Stat??>

<table id="ms\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Flights Delayed</th>

<th>Flights Early</th>

<th>Flights On Time</th>

<th>Flights Full</th>

<th>Flights Half Empty</th>

</tr>

</thread>

<tbody>

<#if Stat.flightsDelayed?? && Stat.flightsEarly?? && Stat.flightsOnTime?? && Stat.flightsFull?? && Stat.flightsHalfEmpty??>

<tr>

<td>${Stat.flightsDelayed}</td>

<td>${Stat.flightsEarly}</td>

<td>${Stat.flightsOnTime}</td>

<td>${Stat.flightsFull}</td>

<td>${Stat.flightsHalfEmpty}</td>

</tr>

</#if>

</tbody>

<#else>

Statistics Not Available.

</#if>

</table>

<div class="fl-r" style="margin:0 24px">

<table id="ft\_table" class="row-focus ta-c t\_tbl fs-n">

<thead>

<tr>

<th>Flight To</th>

<th>Number of Flights</th>

</tr>

</thead>

<tbody>

<#list Stat.flightOriginMap?keys as origin>

<tr>

<td>${origin}</td>

<td>${Stat.flightOriginMap.get(origin)}</td>

</tr>

</#list>

</tbody>

</table>

</div>

<div class="fl\_l" style="margin:0 24px">

<table id="ff\_table" class="row-focus ta-c t\_tbl fs-n">

<thead>

<tr>

<th>Flight From</th>

<th>Number of Flights</th>

</tr>

</thead>

<tbody>

<#list Stat.flightDestinationMap?keys as destination>

<tr>

<td>${destination}</td>

<td>${Stat.flightDestinationMap.get(destination)}</td>

</tr>

</#list>

</tbody>

</table>

</div>

<table id="mpd\_table" class="row-focus ta-c fs-n">

<thead>

<tr>

<th>Most Popular Destination</th>

</tr>

</thread>

<tbody>

<#if Stat.mostPopular??>

<tr>

<td>${Stat.mostPopular}</td>

</tr>

</#if>

</tbody>

</table>

<div class="p-basic"></div>

<script>

jQuery(document).ready(function($) {

$('#ft\_table').tableScroll({height:200});

$('#ff\_table').tableScroll({height:200});

});

</script>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

viewStatistics\_gui.ftl

<html>

<head>

<title>Flight Statistics</title>

<link rel="stylesheet" href="css/airplan\_css.css" />

<script>

function notDone() {

var waitamin = confirm("not yet implemented...click Cancel");

return waitamin;

}

</script>

</head>

<body class="bg-gray">

<div class="bg-header">

<div class="header alt-font" style="background-image:url(imgs/3.png)">

<div class="header-logo">

<img width="176" height="62" alt="AirPlan" src="css/logoairplan.PNG">

</div>

<div class="header-links">

<div class="header-pers fs-xs">

<div class="header-banner">

</div>

<div id="header-user-links" class="header-user-links">

<ul class="menu top-menu header-user-link" id="signin-menu">

<li class="mm-item">

Welcome, \*USERNAME HERE\* &nbsp; &nbsp; <a class="mm-item-link c9" href="staffSignIn.action">Sign Out</a>

</li>

</ul>

</div>

</div>

<ul class="header-menus fs-m menu">

<li class="header-menu mm-item">

<a class="mm-item-link" href="" onclick="return notDone()">Manage User Bookings</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link" href="selectManifest.action?userId=${userId}">View Flight Manifests</a>

</li>

<li class="header-menu mm-item">

<a class="mm-item-link-sel" href="viewStatistics.action?userId=${userId}">View Flight Statistics</a>

</li>

</ul>

</div>

</div>

</div>

<div class="bg">

<div class="wrap">

<div class="content-1024">

<div class="col-1024">

<div class="p-basic"></div>

<fieldset class="col-600" id="corner-rnd">

<legend>Flight Routes</legend>

<form name="form1" method="get" action="/airplan\_repo/viewStatistics.action">

<div class="labels">

<label for="origin">Origin:</label>

<label for="destination">Destination:</label>

</div>

<div class="inputs">

<input name="userId" id="userId" type="hidden" value=${userId} />

<input name="origin" class="textbox-gen" id="origin" type="text" />

<input name="destination" class="textbox-gen" id="destination" type="text" />

<input type="submit" class="positive btn-gen" value="Show Statistics" />

</div>

</form>

</fieldset>

<div class="p-basic"></div>

</div>

</div>

</div>

</div>

<div class="ftr">

<div class="p-27-t">

<div class="col-768">

<p class="fs-xxs">(C)2012 pod10</p>

</div>

</div>

</div>

</body>

</html>

struts.xml

<?xml version="1.0" encoding="UTF-8" ?>

<!DOCTYPE struts PUBLIC

"-//Apache Software Foundation//DTD Struts Configuration 2.0//EN"

"http://struts.apache.org/dtds/struts-2.0.dtd">

<struts>

<constant name="struts.enable.DynamicMethodInvocation" value="false" />

<constant name="struts.devMode" value="true" />

<include file="struts-default.xml"/>

<package name="tutorial" namespace="/" extends="struts-default">

<action name="viewManifest" class="ubc.ece419.pod10.action.staff.ViewManifestAction" method="execute">

<param name="userId">userId</param>

<param name="flightManifest\_flightId">flightManifest\_flightId</param>

<result type="freemarker">/flightManifest.ftl</result>

</action>

<action name="cancelBooking" class="ubc.ece419.pod10.action.user.CancelBookingAction" method="execute">

<param name="userId">userId</param>

<param name="bookingId">bookingId</param>

<result type="freemarker">/cancelBooking.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="confirmCancel" class="ubc.ece419.pod10.action.user.ConfirmCancelAction" method="execute">

<param name="userId">userId</param>

<param name="bookingId">bookingId</param>

<param name="billingId">billingId</param>

<result type="freemarker">/confirmCancel.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="createUserInput" class="ubc.ece419.pod10.action.BaseAction" method="execute">

<result name="success" type="freemarker">/newUser.ftl</result>

</action>

<action name="createUser" class="ubc.ece419.pod10.action.user.CreateUserAction" method="execute">

<param name="name">name</param>

<param name="password">password</param>

<param name="emailAddress">emailAddress</param>

<param name="passwordCopy">passwordCopy</param>

<result name="success" type="redirect-action">

<param name="actionName">searchFlightsGui</param>

<param name="userId">${userId}</param>

</result>

</action>

<action name="signIn" class="ubc.ece419.pod10.action.BaseAction" method="execute">

<result type="freemarker">/signIn.ftl</result>

</action>

<action name="login" class="ubc.ece419.pod10.action.user.LoginAction" method="execute">

<param name="emailAddress">emailAddress</param>

<param name="password">password</param>

<result name="input" type="freemarker">/signIn.ftl</result>

<result name="error" type="freemarker">/signIn.ftl</result>

<result name="success" type="redirectAction">

<param name="actionName">searchFlightsGui</param>

<param name="userId">${userId}</param>

</result>

</action>

<action name="staffSignIn" class="ubc.ece419.pod10.action.BaseAction" method="execute">

<result type="freemarker">/staffSignIn.ftl</result>

</action>

<action name="stafflogin" class="ubc.ece419.pod10.action.staff.StaffLoginAction" method="execute">

<param name="emailAddress">emailAddress</param>

<param name="password">password</param>

<result name="input" type="freemarker">/staffSignIn.ftl</result>

<result name="error" type="freemarker">/staffSignIn.ftl</result>

<result name="success" type="redirectAction">

<param name="actionName">selectManifest</param>

<param name="userId">${userId}</param>

</result>

</action>

<action name="searchFlightsGui" class="ubc.ece419.pod10.action.user.UserSearchFlightsAction" method="getSearchGuiInput">

<param name="userId">userId</param>

<result name="success" type="freemarker">/searchFlights\_gui.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="searchFlightsReturnGui" class="ubc.ece419.pod10.action.user.UserSearchFlightsAction" method="getSearchReturnGuiInput">

<param name="userId">userId</param>

<param name="bookingId">bookingId</param>

<param name="origin">origin</param>

<param name="destination">destination</param>

<result name="success" type="freemarker">/searchFlightsReturn\_gui.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="searchFlights" class="ubc.ece419.pod10.action.user.UserSearchFlightsAction" method="execute">

<param name="userId">userId</param>

<param name="destination">destination</param>

<param name="origin">origin</param>

<param name="departureTime">departureTime</param>

<param name="requiredSeats">requiredSeats</param>

<param name="seatType">seatType</param>

<result name="success" type="freemarker">/searchFlights.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="reserveFlightsGui" class="ubc.ece419.pod10.action.user.UserReserveFlightsAction" method="getReserveGuiInput">

<param name="userId">userId</param>

<param name="flightId">flightId</param>

<param name="seatType">seatType</param>

<result name="success" type="freemarker">/reserveFlights\_gui.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="reserveFlights" class="ubc.ece419.pod10.action.user.UserReserveFlightsAction" method="execute">

<param name="userId">userId</param>

<param name="flightId">flightId</param>

<param name="passengerName">passengerName</param>

<param name="seatType">seatType</param>

<param name="mealPreference">mealPreference</param>

<param name="seatPreference">seatPreference</param>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

<result name="success" type="freemarker">/reserveFlights.ftl</result>

</action>

<action name="searchFlightsLoopGui" class="ubc.ece419.pod10.action.user.UserSearchFlightsAction" method="getSearchGuiInputLoop">

<param name="userId">userId</param>

<param name="bookingId">bookingId</param>

<result name="success" type="freemarker">/searchFlightsLoop\_gui.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="searchFlightsLoop" class="ubc.ece419.pod10.action.user.UserSearchFlightsAction" method="executeLoop">

<param name="bookingId">bookingId</param>

<param name="userId">userId</param>

<param name="destination">destination</param>

<param name="origin">origin</param>

<param name="departureTime">departureTime</param>

<param name="requiredSeats">requiredSeats</param>

<param name="seatType">seatType</param>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

<result name="success" type="freemarker">/searchFlightsLoop.ftl</result>

</action>

<action name="reserveFlightsLoopGui" class="ubc.ece419.pod10.action.user.UserReserveFlightsAction" method="getReserveGuiInputLoop">

<param name="userId">userId</param>

<param name="bookingId">bookingId</param>

<param name="flightId">flightId</param>

<param name="seatType">seatType</param>

<result name="success" type="freemarker">/reserveFlightsLoop\_gui.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="reserveFlightsLoop" class="ubc.ece419.pod10.action.user.UserReserveFlightsAction" method="executeLoop">

<param name="userId">userId</param>

<param name="bookingId">bookingId</param>

<param name="flightId">flightId</param>

<param name="passengerName">passengerName</param>

<param name="seatType">seatType</param>

<param name="mealPreference">mealPreference</param>

<param name="seatPreference">seatPreference</param>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

<result name="success" type="freemarker">/reserveFlightsLoop.ftl</result>

</action>

<action name="makePaymentGui" class="ubc.ece419.pod10.action.user.UserReserveFlightsAction" method="getBillingGuiInput">

<param name="userId">userId</param>

<param name="bookingId">bookingId</param>

<result name="success" type="freemarker">/makePayment\_gui.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="makePayment" class="ubc.ece419.pod10.action.user.UserReserveFlightsAction" method="makePayment">

<param name="userId">userId</param>

<param name="bookingId">bookingId</param>

<param name="subtotal">subtotal</param>

<param name="isPayByPoints">isPayByPoints</param>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

<result name="success" type="freemarker">/makePayment.ftl</result>

</action>

<action name="viewBooking" class="ubc.ece419.pod10.action.user.ViewBookingAction" method="execute">

<param name="bookingId">bookingId</param>

<param name="userId">userId</param>

<result type="freemarker">/viewBooking.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="viewBookingList" class="ubc.ece419.pod10.action.user.ViewBookingListAction" method="execute">

<param name="userId">userId</param>

<result type="freemarker">/viewBookingList.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="viewTransactions" class="ubc.ece419.pod10.action.user.ViewTransactionHistoryAction" method="execute">

<param name="userId">userId</param>

<result type="freemarker">/transactions.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<!--

<action name="reservationPayment" class="">

<param name="userId">userId</param>

<param name="bookingId">bookingId</param>

<param name="billingInfo">billingInfo</param>

<param name="creditCardNum">creditCardNum</param>

<param name="cardHolderName">cardHolderName</param>

<param name="loyaltyPoints">loyaltyPoints</param>

</action>

-->

<action name="selectManifest" class="ubc.ece419.pod10.action.staff.SelectManifestAction" method="execute">

<param name="userId">userId</param>

<param name="message">message</param>

<param name="selectionList">selectionList</param>

<result type="freemarker">/selectManifest.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/404NotFoundError.ftl</result>

</action>

<action name="\*">

<result type="freemarker">/404NotFoundError.ftl</result>

</action>

<action name="boardingPass" class="ubc.ece419.pod10.action.user.ViewBoardingPassAction" method="execute">

<param name="userId">userId</param>

<param name="bookingId">bookingId</param>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

<result name="success" type="freemarker">/boardingPass.ftl</result>

</action>

<action name="viewUpcomingFlights" class="ubc.ece419.pod10.action.user.ViewBoardingPassAction" method="input">

<param name="userId">userId</param>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

<result name="success" type="freemarker">/upcomingFlights.ftl</result>

</action>

<action name="modifyTicket" class="ubc.ece419.pod10.action.user.ModifyTicketAction" method="execute">

<param name="ticketId">ticketId</param><param name="passengerName">passengerName</param>

<param name="seatPreference">seatPreference</param>

<param name="mealPreference">mealPreference</param>

<param name="userId">userId</param>

<result name="error" type="redirect">/InternalServerError.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="success" type="freemarker">/modifyTicket.ftl</result>

</action>

<action name="modifyTicketGui" class="ubc.ece419.pod10.action.user.ModifyTicketAction" method="getModifyTicketGuiInput">

<param name="ticketId">ticketId</param><param name="userId">userId</param>

<param name="userId">userId</param>

<result name="error" type="redirect">/InternalServerError.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="success" type="freemarker">/modifyTicket\_gui.ftl</result>

</action>

<action name="viewStatistics" class="ubc.ece419.pod10.action.staff.ViewStatisticsAction" method="execute">

<param name="userId">userId</param>

<result name="error" type="redirect">/InternalServerError.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="success" type="freemarker">/viewStatistics.ftl</result>

</action>

<action name="staffCancelBooking" class="ubc.ece419.pod10.action.staff.StaffCancelBookingAction" method="execute">

<param name="userId">userId</param>

<param name="customerId">customerId</param>

<param name="bookingId">bookingId</param>

<result type="freemarker">/staffCancelBooking.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="staffConfirmCancel" class="ubc.ece419.pod10.action.staff.StaffConfirmCancelAction" method="execute">

<param name="userId">userId</param>

<param name="customerId">customerId</param>

<param name="bookingId">bookingId</param>

<param name="billingId">billingId</param>

<result type="freemarker">/staffConfirmCancel.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="staffModifyTicket" class="ubc.ece419.pod10.action.staff.StaffModifyTicketAction" method="execute">

<param name="ticketId">ticketId</param><param name="passengerName">passengerName</param>

<param name="seatPreference">seatPreference</param>

<param name="mealPreference">mealPreference</param>

<param name="userId">userId</param>

<param name="customerId">customerId</param>

<result name="error" type="redirect">/InternalServerError.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="success" type="freemarker">/staffModifyTicket.ftl</result>

</action>

<action name="staffModifyTicketGui" class="ubc.ece419.pod10.action.staff.StaffModifyTicketAction" method="getModifyTicketGuiInput">

<param name="ticketId">ticketId</param><param name="userId">userId</param><param name="customerId">customerId</param>

<result name="error" type="redirect">/InternalServerError.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="success" type="freemarker">/staffModifyTicket\_gui.ftl</result>

</action>

<action name="staffViewBooking" class="ubc.ece419.pod10.action.staff.StaffViewBookingAction" method="execute">

<param name="bookingId">bookingId</param>

<param name="userId">userId</param>

<param name="customerId">customerId</param>

<result type="freemarker">/staffViewBooking.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

<action name="staffViewBookingList" class="ubc.ece419.pod10.action.staff.StaffViewBookingListAction" method="execute">

<param name="userId">userId</param>

<param name="customerId">customerId</param>

<result type="freemarker">/staffViewBookingList.ftl</result>

<result name="login" type="redirect">/signIn.ftl</result>

<result name="error" type="redirect">/InternalServerError.ftl</result>

</action>

</package>

</struts>

web.xml

<!DOCTYPE web-app PUBLIC

"-//Sun Microsystems, Inc.//DTD Web Application 2.3//EN"

"http://java.sun.com/dtd/web-app\_2\_3.dtd" >

<web-app>

<display-name>Archetype Created Web Application</display-name>

<filter>

<filter-name>struts2</filter-name>

<filter-class>org.apache.struts2.dispatcher.FilterDispatcher</filter-class>

</filter>

<filter-mapping>

<filter-name>struts2</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

<welcome-file-list>

<welcome-file>index.jsp</welcome-file>

<welcome-file>index.html</welcome-file>

</welcome-file-list>

</web-app>